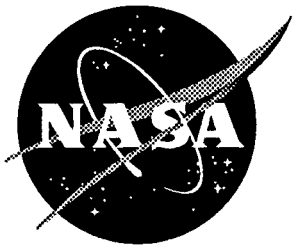
NASA Technical Memorandum 110166



The Transition of a Real-Time Single-Rotor Helicopter Simulation Program to a Supercomputer

Debbie Martinez
Langley Research Center, Hampton, Virginia

October 1995

National Aeronautics and
Space Administration
Langley Research Center
Hampton, Virginia 23681-0001

TABLE OF CONTENTS

LIST OF ACRONYMS ____ iii

SUMMARY ____ 1

INTRODUCTION ____ 1

SYMBOLS ____ 3

PROBLEM DESCRIPTION ____ 13

PROGRAM ORGANIZATION ____ 15

COMMON BLOCK DESCRIPTIONS ____ *17*
USER COMMON BLOCKS ____ 17
SOFTWARE INTERFACE COMMON BLOCKS ____ 25

SOFTWARE INTERFACE ARRAYS ____ *26*
Analog-to-Digital Converter ____ 26
Digital-to-Analog Converter ____ 26
Synchro ____ 27
Input Discretes ____ 28
Output Discretes ____ 28
Function Sense Switches ____ 29
Mode Control Panel Switches ____ 29
White Lights ____ 30
Red Lights ____ 31

EXTERNAL SUBROUTINE DESCRIPTIONS ____ *32*
REAL-TIME SUPERVISOR SUBROUTINE DESCRIPTIONS ____ 32
CALLIGRAPHIC RASTER DISPLAY SYSTEM (CRDS) SUBROUTINES ____ 32
COMPUTER-GENERATED IMAGE (CGI) SYSTEM SUBROUTINES ____ 32
REAL-TIME SYSTEM CONSOLE (RTSC) SUBROUTINES ____ 33
REAL-TIME SYSTEM MAIN SUBROUTINES ____ 34
SUPPLEMENTARY SUBROUTINE DESCRIPTIONS ____ 35

S-61 REAL-TIME PROGRAM ____ *36*
S-61 REAL-TIME APPLICATION MAIN PROGRAM FORTRAN LISTING *(S61MB.F)* ____ 36
S-61 REAL-TIME SUBROUTINE DESCRIPTIONS ____ 47

PROGRAM USAGE ____ 51

OPERATION PROCEDURES ____ *51*
PROGRAM SETUP ____ *52*
COCKPIT CHECKOUT ____ *54*
CONSOLE OPERATION PROCEDURE ____ *54*

PROGRAM VALIDATION ____ 56

CONCLUDING REMARKS ____ 56

APPENDIX A - OPERATIONAL PROCEDURES ____ 59

APPENDIX B - MAKEBATCH SCRIPT LISTING 60

APPENDIX C - HIGH LEVEL S-61 PROGRAM FLOW CHART 61

APPENDIX D - COMMON BLOCK FORTRAN LISTING 62

APPENDIX E - BULKDAT FORTRAN LISTING 73

REFERENCES 80

TABLE I.A. - S-61 PROGRAM DAILY STATIC CHECK 81

TABLE I.B. - S-61 PROGRAM FULL VALIDATION STATIC CHECK 82

TABLE II.A. - DYNAMIC CHECK CASE (IDYNCK 200) 83

TABLE II.B. - DYNAMIC CHECK CASE (IDYNCK 100) 89

TABLE III.A. - STATIC COCKPIT INSTRUMENT CHECK 95

TABLE III.B. - CONTROL DEFLECTION CHECK 96

TABLE III.C. - CONTROL DEFLECTION VOLTAGE SCALE 96

FIGURE 1 - VISUAL MOTION SIMULATOR 97

FIGURE 2 - DENVER STAPLETON AIRPORT DATA BASE 98

FIGURE 3 - S-61 ELECTRONIC ATTITUDE DIRECTIONAL INDICATOR 99

FIGURE 4 - REAL-TIME CONTROL CONSOLE 100

FIGURE 5 - REAL-TIME SYSTEM CONSOLE PC 101

FIGURE 6 - S-61 TOUCH SCREEN CONSOLE FORMAT 102

LIST OF ACRONYMS

ADC	Analog-to-Digital Converter
ARTSS	Advanced Real-Time Simulation System
CAMAC	Computer Automated Measurement and Control
CDC	Control Data Corporation
CGI	Computer Generated Image
CLITE	Console lights
CMP	Combat Monitor Program
CONVEXOS	CONVEX UNIX Operating System
CRDS	Calligraphic Raster Display System
DAC	Digital-to-Analog Converter
DCS	Dynamic Coordinate System
DOF	Degrees-Of-Freedom
EADI	Electronic Attitude Directional Indicator
ECC	Engineering Control Console
FAA	Federal Aviation Administration
FOV	Field-Of-View
FSCS	Flight Simulation Computing System
FSS	Function Sense Switches
IDIS	Input discretes
I/O	Input/Output
LaRC	Langley Research Center
MB	Motion Base
MCP	Mode Control Panel
MIDI	Musical Instrument Digital Interface
MRT	Moveable Real-Time
NOS	Network Operating System
ODIS	Output discretes
PC	Personal Computer
PDP	Programmable Display Pushbutton
RLITE	Red lights
RTCC	Real-Time Control Console
RTSC	Real-Time System Console
SAS	Stability Augmentation System
SCO	Santa Cruz Operation
SHIP	Serial Highway Interface Processeor
SI	International System of Units
SMM	Symbolic Memory Map
SRT	Synchronized Real-Time
VMS	Visual Motion Simulator
WLITE	White lights

The Transition of a Real-Time Single-Rotor Helicopter Simulation Program to a Supercomputer

By
Debbie Martínez
Langley Research Center
Hampton, VA 23681-0001

SUMMARY

This report presents the conversion effort and results of a real-time flight simulation application transition to a CONVEX supercomputer. Included is a detailed description of the conversion process and a brief description of the Langley Research Center's (LaRC) flight simulation application program structure.

Highlighted is a detailed description of the user and software interface COMMON blocks, software interface arrays, a brief description of various external routines, program usage, and a FORTRAN listing of the converted real-time helicopter application main program followed by a detailed description of its subroutines.

INTRODUCTION

This paper describes the conversion effort of a real-time flight simulation program application transition from the Control Data Corporation (CDC) CYBER 175 computer to the current CONVEX Computer Corporation C3800 supercomputer. This document may also prove useful to the novice real-time simulation user or researcher who would like an understanding of a typical LaRC flight simulation application program.

For this document, the expressions "real-time flight simulation" or "real-time application program" relate to the computer program execution of a modeled dynamic process of a particular rotorcraft that occurs in the real-world time (i.e., not faster or slower). Therefore, program execution represents or simulates the real dynamic phenomenon of such a rotorcraft as it occurs. The process is characterized by using an aircraft cockpit, hardware, pilot, and supercomputer all in a closed loop system.

This simulation program may be configured to represent a Sikorsky S-61 helicopter, a five-blade, single-rotor, commercial passenger-type helicopter or an Army Cobra helicopter, either the AH-1G or AH-1S model. Since this simulation program is most frequently used in a Sikorsky

S-61 configuration; the program will also be referred to as the "S-61 real-time application program". This simulation program originally was developed for the study of a single-rotor helicopter documented in reference 1. It was then modified to conduct two additional studies. One of these studies involved the development and evaluation of various computer-generated displays and instruments for aiding pilots during flight, especially during approach and landing. A second study concerned pilot workload and the development of route structures and air traffic control procedures. This study concentrated on instrument flight in congested short-haul markets for an intracity helicopter passenger service and is documented in reference 2.

Three years later, the simulation program was combined with visual, motion and aural cues to conduct an empirical comparison study between a fixed-base and a moving-base simulation, documented in reference 3. Throughout the years that followed, the simulation program was used to conduct several other studies supporting the U.S. Air Force, U.S. Army, U.S. Navy, and the Federal Aviation Administration (FAA) research.

Currently, no formal research is being conducted with this simulation program but it has been used to gain experience in real-time flight simulation programming. The U.S. Army has shown an interest in using this simulation program to conduct helicopter drop model test flight training.

SYMBOLS

Measurements, calculations, and programming were made in the U.S. Customary Units. They are presented herein in the International System of Units (SI) with the equivalent values in the U.S. Customary Units given parenthetically. The symbols list is adapted from reference 1.

Symbol	FORTRAN Variable	Description
A_{SPD}	ASPD	Indicated airspeed, kilometers/hr (knots)
A_{OS}	AOS	Main rotor collective pitch, radians
A_{OSC}	AOSC	Collective pitch control input, radians
A_{1S}	A1S	Main rotor lateral cyclic pitch, radians
A_{1SC}	A1SC	Lateral cyclic pitch control input, radians
A_{1SCS}	A1SCS	SAS lateral cyclic pitch control input, radians
ADV_{P1}	ADVP1	Aerodynamic variable parameter, per meter^2 (per foot^2)
ADV_{P2}	ADVP2	Aerodynamic variable parameter, per second
a_X,a_Y,a_Z	AX,AY,AZ	Linear acceleration along X, Y, and Z body axis, respectively, meters/s^2 (ft/s^2)
B_{1S}	B1S	Main rotor longitudinal cyclic pitch, radians
B_{1SC}	B1SC	Longitudinal cyclic pitch control input, radians
B_{1SCS}	B1SCS	SAS longitudinal cyclic pitch control input, radians
$C_{D\psi Y}$	CDSIY	Main rotor drag coefficient at ψ_R and Y
$C_{L\psi Y}$	ClSIY	Main rotor lift coefficient at ψ_R and Y
C_P	CP	Power coefficient
C_Q	CQ	Torque coefficient
C_T	CT	Thrust coefficient

Symbol	FORTRAN Variable	Description
C_{X_w}	CXW	Wing drag coefficient
C_{X1}	CX1	Fuselage drag coefficient as a function of sideslip
C_{X2}	CX2	Fuselage drag coefficient as a function of angle of attack
C_{YF}	CYF	Fuselage side-force coefficient
C_{YTR}	CYTR	Tail rotor side-force coefficient
C_{YVS}	CYVS	Vertical-stabilizer side-force coefficient
C_{ZF}	CZF	Fuselage vertical-force coefficient
C_{ZHS}	CZHS	Horizontal-stabilizer vertical-force coefficient
C_{ZW}	CZW	Wing vertical-force coefficient
C_{lF1}	CLF1	Fuselage rolling-moment coefficient
C_{lF2}	CLF2	Fuselage roll-rate damping coefficient, kilogram-meters (slug-ft)
C_{mF1}	CMF1	Fuselage pitching-moment coefficient
C_{mF2}	CMF2	Fuselage pitch-rate damping coefficient, kilogram-meters (slug-ft)
C_{nF1}	CNF1	Fuselage yawing-moment coefficient
C_{nF2}	CNF2	Fuselage yaw-rate damping coefficient, kilogram-meters (slug-ft)
D_W	DW	Rotor downwash coefficient
D_X	DX	Longitudinal distance from center of gravity to main rotor shaft (positive rearward of shaft), meters (feet)
D_{XHS}	DXHS	Longitudinal distance from horizontal stabilizer to main rotor shaft (positive rearward of shaft), meters (feet)
D_{WTR}	DXTR	Longitudinal distance from tail rotor to main rotor shaft (positive rearward of shaft), meters (feet)

Symbol	FORTRAN Variable	Description
D_{XVS}	DXVS	Longitudinal distance from vertical stabilizer to main rotor shaft (positive rearward of shaft), meters (feet)
D_{XW}	DXW	Longitudinal distance from wing to main rotor shaft (positive rearward of shaft), meters (feet)
D_Z	DZ	Vertical distance from center of gravity to main rotor hub (positive upward to hub), meters (feet)
D_{ZTR}	DZTR	Vertical distance from tail rotor to main rotor hub (positive upward to hub), meters (feet)
D_{ZVS}	DZVS	Vertical distance from vertical-stabilizer to main rotor hub (positive upward to hub), meters (feet)
D_{ZW}	DZW	Vertical distance from wing to main rotor hub (positive upward to hub), meters (feet)
E	E	Earth coordinate in east direction, meters (feet)
$\dot{E}$	EDOT	Velocity component in east direction, meters/sec (ft/sec)
e	EFH	Flapping hinge offset, meters (feet)
F_{MRS}	FMRS	Main rotor hub moment parameter, kilogram-meters^2 (slug-ft^2)
F_{XR}	FXR	Main rotor force along X body axis, newtons (pounds)
F_{YR}	FYR	Main rotor force along Y body axis, newtons (pounds)
G_{EH}	GEH	Main rotor ground effect due to height
G_{EV}	GEV	Main rotor ground effect due to forward velocity
G_{RWT}	GRWT	Vehicle gross weight, kilograms (pounds)
g	G	Acceleration due to gravity, meters/sec^2 (ft/sec^2)
h	H	Earth vetical coordinate, meters (feet)
$\dot{h}$	HDOT	Velocity component in vertical direction, meters/sec (ft/sec)

Symbol	FORTRAN Variable	Description
h_D	HDOTD	Desired velocity component in vertical direction, meters/sec (ft/sec)
I_B	IB	Blade moment of inertia, kilogram-meters^2 (slug-ft^2)
I_{HS}	IHS	Horizontal-stabilizer built-in incidence angle, radians
I_R	IR	Rotor moment of inertia inboard of flapping hinge, kilogram-meters^2 (slug-ft^2)
I_{ROT}	IROT	Main rotor moment of inertia, kilogram-meters^2 (slug-ft^2)
I_{VS}	IVS	Vertical-stabilizer built-in incidence angle, radians
I_W	IW	Wing built-in incidence angle, radians
I_{XX}	IXX	Moment of inertia about X body axis, kilogram-meters^2 (slug-ft^2)
I_{YY}	IYY	Moment of inertia about Y body axis, kilogram-meters^2 (slug-ft^2)
I_{ZZ}	IZZ	Moment of inertia about Z body axis, kilogram-meters^2 (slug-ft^2)
K_{AO}, K_{AB1}	CKAO,CKAB1	Main rotor constants for flap angle calculation
K_{AB2}, K_{AB3}	CKAB2,CKAB3	Main rotor constants for flap angle calculation
K_D,K_{DL},K_L	CKD,CKDL,CKL	Area constants used to determine main rotor drag and lift forces, meters^2 (ft^2)
K_Q,K_{QL},K_M	CKQ,CKQL,CKM	Volume constants used to determine main rotor toque and moment, meters^3 (ft^3)
K_{E1}	KE1	Constant used in engine torque calculation, kilogram-meters^2/sec (slug-ft^2/sec)
$K_{R/F}$	CKRF	Rotor downwash fraction acting on fuselage and wing
K_{TR1}	CKTR1	Ratio of elevator area to horizontal-stabilizer area

Symbol	FORTRAN Variable	Description
K_{TR2}	CKTR2	Constant used in calculation of Y_{TR}, newtons (pounds)
K_{WIWM}	CKWIWM	Wing downwash coefficient
L_A	LA	Total aerodynamic rolling moment, newton-meters (pound-ft)
L_F	LF	Fuselage rolling moment, newton-meters (pound-ft)
L_{MR}	LMR	Main rotor lift force, newtons (pounds)
L_{RH}	LRH	Main rotor hub rolling moment, newton-meters (pound-ft)
M_A	MA	Total aerodynamic pitching moment, newton-meters (pound-ft)
M_F	MF	Fuselage pitching moment, newton-meters (pound-ft)
M_{RH}	MRH	Main rotor hub pitching moment, newton-meters (pound-ft)
M_{TIP}	MTIP	Main rotor tip Mach number
m	MASS	Vehicle mass, kilograms (slugs)
N	N	Earth coordinate in north direction, meters (feet)
$\dot{N}$	NDOT	Velocity component in north direction, meters/sec (ft/sec)
N_A	NA	Total aerodynamic yawing moment, newton-meters (pound-ft)
N_{AZ}	NAZ	Number of rotor azimuth stations
N_B	NB	Number of rotor blades
N_F	NF	Fuselage yawing moment, newton-meters (pound-ft)
N_{RAD}	NRAD	Number of blade radial stations
p	P	Roll rate, radians/sec
$\dot{p}$	PDOT	Roll acceleration, radians/sec^2
Q_E	QE	Engine torque, newton-meters (pound-ft)

Symbol	FORTRAN Variable	Description
Q_{ED}	QED	Desired engine torque, newton-meters (pound-ft)
Q_{MAX}	QMAX	Maximum engine torque, newton-meters (pound-ft)
Q_{MR}	QMR	Main rotor torque, newton-meters (pound-ft)
q	Q	Pitch rate, radians/sec
$\dot{q}$	QDOT	Pitch acceleration, radians/sec^2
$\bar{q}_L$	$\bar{q}$L	Lateral dynamic pressure, newtons/meter^2 (pounds/foot^2)
$\bar{q}_V$	$\bar{q}$V	Vertical dynamic pressure, newtons/meter^2 (pounds/foot^2)
R_B	RB	Main rotor blade radius, meters (feet)
r	R	Yaw rate, radians/sec
$\dot{r}$	RDOT	Yaw acceleration, radians/sec^2
S_{HP}	SHP	Shaft horsepower, watts (horsepower)
S_{XF1}	SXF1	Fuselage lateral-force area, meters^2 (ft^2)
S_{XF2}	SXF2	Fuselage vertical-force area, meters^2 (ft^2)
S_{XW}	SXW	Wing longitudinal-force area, meters^2 (ft^2)
S_{YF}	SYF	Fuselage lateral-force area, meters^2 (ft^2)
S_{YVS}	SYVS	Vertical-stabilizer force area, meters^2 (ft^2)
S_{ZF}	SZF	Fuselage vertical-force area, meters^2 (ft^2)
S_{ZHS}	SZHS	Horizontal-stabilizer force area, meters^2 (ft^2)
S_{ZW}	SZW	Wing vertical-force area, meters^2 (ft^2)
t	T	Time, seconds

Symbol	FORTRAN Variable	Description
$U_{P\psi Y}$	UPSIY	Blade element velocity perpendicular to blade span axis and to UTSIY at ψ_R and Y, meters/sec (ft/sec)
$U_{T\psi Y}$	UTSIY	Blade element velocity perpendicular to blade span axis and to rotor rotation axis at ψ_R and Y, meters/sec (ft/sec)
u	U	Linear velocity along X body axis, meters/sec (ft/sec)
$\dot{u}$	UDOT	Linear acceleration along X body axis, meters/sec^2 (ft/sec^2)
V_S	VS	Speed of sound, meters/sec (ft/sec)
V_T	VT	Total velocity of air through main rotor, meters/sec (ft/sec)
V_{TR}	VTR	Tail velocity along Y body axis, meters/sec (ft/sec)
V_{lF}	VLF	Fuselage rolling-moment volume parameter, meters^3 (ft^3)
V_{mF}	VMF	Fuselage pitching-moment volume parameter, meters^3 (ft^3)
V_{nF}	VNF	Fuselage yawing-moment volume parameter, meters^3 (ft^3)
v	V	Linear velocity along Y body axis, meters/sec (ft/sec)
$\dot{v}$	VDOT	Linear acceleration along Y body axis, meters/sec^2 (ft/sec^2)
W_F	WF	Main rotor inflow distribution at any radial station
W_{HS}	WHS	Tail velocity along Z body axis, meters/sec (ft/sec)
W_{IF}	WIF	Main rotor inflow redistribution as function of velocity, meters/sec (ft/sec)
W_{IM}	WIM	Main rotor mean inflow velocity, meters/sec (ft/sec)
W_{IWM}	WIWM	Wing downwash velocity, meters/sec (ft/sec)
$W_{I\psi Y}$	WISIY	Main rotor local inflow velocity, meters/sec (ft/sec)
w	W	Linear velocity along Z body axis, meters/sec (ft/sec)
$\dot{w}$	WDOT	Linear acceleration along Z body axis, meters/sec^2 (ft/sec^2)

Symbol	FORTRAN Variable	Description
X_A	XA	Total aerodynamic force along X body axis, newtons (pounds)
X_{AOS}	XAOS	Cockpit collective stick input (positive up), centimeters (in)
X_{CS}	XCS	Cockpit cyclic stick longitudinal input (positive forward), centimeters (in)
X_F	XF	Fuselage aerodynamic force along X body axis, newtons (pounds)
X_{TR}	XTR	Cockpit tail rotor pedal input (positive right), centimeters (in)
X_W	XW	Wing aerodynamic force along X body axis, newtons (pounds)
Y	Y	Main rotor radial distance (measured from hinge), meters (feet)
Y_A	YA	Total aerodynamic force along Y body axis, newtons (pounds)
Y_{CS}	YCS	Cockpit cyclic stick lateral input (positive right), centimeters (in)
Y_F	YF	Fuselage aerodynamic force along Y body axis, newtons (pounds)
Y_{PE}	YPE	Main rotor radial station plus hinge offset, meters (feet)
Y_{TR}	YTR	Tail rotor force along Y body axis, newtons (pounds)
Y_{TW}	YTW	Blade twist at any radial station, radians
Y_{VS}	YVS	Vertical-stabilizer force along Y body axis, newtons (pounds)
Z_A	ZA	Total aerodynamic force along Z body axis, newtons (pounds)
Z_F	ZF	Fuselage aerodynamic force along Z body axis, newtons (pounds)
Z_{HS}	ZHS	Horizontal-stabilizer force along Z body axis, newtons (pounds)
Z_W	ZW	Wing aerodynamic force along Z body axis, newtons (pounds)
α_F	ALFF	Fuselage angle of attack, radians or degrees
α_{HS}	ALFHS	Horizontal-stabilizer angle of attack, radians

Symbol	FORTRAN Variable	Description
α_W	ALFW	Wing angle of attack, radians
α_{OSS}	AOSS	Main rotor coning angle, radians
α_{1SS}	A1SS	Main rotor longitudinal flap angle, radians
$\dot{\alpha}_{1SS}$	AD1SS	Main rotor longitudinal flap rate, radians/sec
$\alpha_{\psi y}$	ALFSIY	Main rotor blade angle of attack at ψ_R and Y, radians
β_F	BETF	Fuselage angle of sideslip, radians or degrees
β_{VS}	BTVS	Vertical-stabilizer angle of sideslip, radians
β_{1SS}	B1SS	Main rotor lateral flap angle, radians
$\dot{\beta}_{1SS}$	BD1SS	Main rotor lateral flap rate, radians/sec
β_ψ	BETSI	Main rotor total blade flap angle at ψ_R, radians
$\dot{\beta}_\psi$	BDTSI	Main rotor total blade flap rate at ψ_R, radians/sec
Δt	DELTA	Time interval, seconds
δ_E	DELE	Horizontal-stabilizer deflection angle, radians or degrees
θ	THETA	Fuselage pitch angle, radians
$\dot{\theta}$	THETDT	Rate of change of pitch angle, radians/sec
θ_{TR}	THTR	Tail rotor pitch angle, radians
θ_{TRC}	THTRC	Tail rotor pitch control input, radians
θ_{TRCS}	THTRCS	SAS tail rotor pitch control input, radians
$\theta_{\psi Y}$	THSIY	Main rotor blade pitch angle at ψ_R and Y, radians
μ	MU	Main rotor tip-speed ratio

Symbol	FORTRAN Variable	Description
ρ	RHO	Air density, kilogram/meter^3 (slug/ft^3)
ϕ	PHI	Fuselage roll angle, radians
$\dot{\phi}$	PHID	Rate of change of roll angle, radians/sec
$\phi_{\psi Y}$	PHIY	Main rotor blade inflow angle at ψ_R and Y, radians
ψ	PSI	Fuselage yaw angle, radians
$\dot{\psi}$	PSIDT	Rate of change of yaw angle, radians/sec
ψ_R	PSIR	Main rotor blade azimuth angle, radians
Ω	OMEG	Main rotor angular velocity, radians/sec
$\dot{\Omega}$	OMEGDT	Main rotor angular acceleration, radians/sec^2
Ω_D	OMEGD	Desired main rotor angular velocity, radians/sec

Dot over symbol denotes time derivative.

PROBLEM DESCRIPTION

With the acquisition of the new CONVEX supercomputers, all flight simulation application programs required conversion to the new Flight Simulation Computing System (FSCS). The Sikorsky S-61 helicopter real-time simulation program was among the application programs that were converted from the CDC CYBER 175 60-bit word-length machine to the CONVEX 64-bit word-length machine. This section describes the major changes that were implemented within the S-61 program source code to assure proper conversion to the new FSCS using CONVEX UNIX Operating System (CONVEXOS), the CONVEX FORTRAN 77 compiler and the newly designed version of the Real-Time Supervisor (Supervisor).

After the obvious first step of transferring the program source code to the CONVEX machine, the S-61 program source code file was broken into its individual components (i.e., COMMON blocks, subroutines, and functions) by '*rcs*' and '*make*' utilities to provide a reusable and maintainable source code. The *'make'* utility provides a simple mechanism for updating a variety of files. It accomplishes this function by comparing the modification dates and times for the target files with those of the dependent files. If any dependent file is more recent than the target file, then a specified series of commands (described within a *modification* file) is executed.

Real-time utilities '*udsplit*' and '*makemake*' were used to perform the initial splitting of the S-61 source file and the initial generation of a *'makefile'*, respectively. The *'makefile'* was then further tailored to include libraries and other options.

Previous real-time application programs required the use of the CDC Network Operating System (NOS) overlay structure to manage the extremely limited memory (~2 megabytes) on the CYBER 175 machine. Due to the current CONVEX abundant memory space (512 megabytes), the overlay structure is no longer required. Therefore, previously used overlays were removed and replaced with subroutine call statements. Consequently, the control structure of the main subroutine 's61mb.f' was completely redesigned and reprogrammed.

The following lists the major modifications that were made to the S-61 source code in order to compile correctly under CONVEX FORTRAN F77 compiler. Previously used filename equivalencing, buffer length specification, and maximum record length specification are now performed differently, e.g., the OPEN statement. All CONVEX FORTRAN Input/Output (I/O) routines specify a valid unit number. Arrays are declared before their reference in an EQUIVALENCE statement. NOS boolean type were replaced by CONVEX FORTRAN logical type. Multiple assignment statements were placed on individual lines.

In compiling the source code the *-p8* option is used to achieve 64-bit representation (the default compilation is 32-bits). In order for the code to execute in real-time, the program must be completely loaded into memory and must be made non-swappable. This process is done with the loader *pre-paged* (-Eprepage) and *non-swap* (-Enonswap) options activated, and all necessary Supervisor and utility libraries are loaded (see Appendix A).

Two nonreal-time Supervisor libraries were developed to provide a batch run capability. This capability allows the program to run without time constraints. These libraries differ from the normal processing real-time libraries in that they allow debugging and analysis without the use of the real-time hardware. A separate script file is used by the S-61 real-time application program to load the batch library (see Appendix B). Script files are C shell programs that may be programmed to invoke and organize UNIX commands for the CONVEXOS.

The bit numbering on the CONVEX machine begins with zero at the rightmost, least significant bit, which is opposite to the CDC Cyber 175. All references to the hardware interface with the Visual Motion Simulator (VMS) (see figure 1) were modified to reflect the changed discrete bit format. Refer to the section on Software Interface Arrays for further details.

Since the hardware and software environment of the real-time simulation has been upgraded, most Supervisor utilities have been completely restructured, redesigned, and reprogrammed. Major efforts were made by the system developers to make the transition of the Supervisor to the new FSCS as transparent to the user as possible. However, wherever applicable within the S-61 real-time application program, user interfaces to the Supervisor utilities were changed, added or removed. After numerous real-time sessions, extensive debugging, and software verification; the S-61 real-time application program was converted to the new FSCS.

In the process of conversion to the new FSCS, other significant improvements were made to the S-61 real-time application program. These include: a quaternion coordinate system, a visual scene displayed by means of four display systems that are interfaced with the Computer Generated Image (CGI) system[1] (see figure 2); the addition of a new Electronic Attitude Directional Indicator (EADI) using the Calligraphic Raster Display System[2] (CRDS) (see figure 3); interface driver to a state-of-the-art real-time sound simulator; the capability to collect data that can be replayed through an external Combat Monitor Program (CMP) to monitor a previous test flight; and an improved Stability Augmentation System (SAS) algorithm.

Future research utilizing the new and improved Sikorsky S-61 helicopter flight simulation program may include a drop model test flight training study.

---

[1] Evans & Sutherland special purpose image generating mainframe (CT6) connected to a general purpose computer (Gould 32/6781) for communication and control.

[2] Three Eagle 1000 units from Terabit Computer Engineering.

PROGRAM ORGANIZATION

LaRC's flight simulation application program design is due to the unique Advanced Real-Time Simulation System (ARTSS) configuration. ARTSS consist of two dedicated high-speed mainframe supercomputers (CONVEX C3840 and C3830 machines) which are connected to an array of simulation sites via a fiber optic digital data (50 megabits/second) network known as the Computer Automated Measurement and Control (CAMAC).

Each simulation application program requires the use of the Supervisor to provide interfacing with the UNIX operating system and the ARTSS. The Supervisor main functions are self-initialization, coordination of I/O to and from the special hardware, packing and unpacking real-time data, and passing buffer addresses to tables used by the simulation processors. Supervisor also provides two nonreal-time supervisor libraries used for debugging and analysis where access to the real-time hardware is not required.[3] The Supervisor reserves the address space of 0xb0000000 to 0xbfffffff for shared memory usage and the real-time loader starts loading of the application program in the address space of 0x80000000 to 0x8fffffff.

The simulation application program must convey to the Supervisor the information necessary to build the required I/O communication tables to access the appropriate simulator site hardware.
The S-61 real-time application *CONFIG.F* establishes all necessary hardware requirements through a call to the Supervisor *RTBUFDF* subroutine.

The Supervisor manages the real-time data storage and retrieval capability. The application user can dynamically control which variables are to be recorded and request that data be read back into the correct central memory addresses. The S-61 real-time application *INIT.F* subroutine establishes the memory locations for both the file and the variable address table through a call to the Supervisor *FILEARR* subroutine.

Initially, a simulation application program must declare all necessary COMMON blocks (user defined and system defined), then calls to various routines are made in order to establish the simulator site hardware interface[4] and real-time disk storage as described above.

Once all system interfacing has been established (*CONFIG* call) and initialized (*INIT* call), the application program proceeds to set up a recovery and resume address location (*RECADD* call) in

---

[3] Details of the Real-Time Supervisor capabilities may be found in a December, 1993 internal LaRC preliminary document entitled "Real-Time Supervisor User's Guide".

[4] All required real-time hardware must be scheduled (or configured) prior to actual execution of the application program.

order to recover and restart the program if any errors are detected by the Supervisor[5]. The program follows to set up the lost-time synchronization recovery (*LOSTIME* call) and enters the Synchronized Real-Time (SRT) program state (*RTSRT* call) to proceed with real-time operations.

The S-61 program structure (like most real-time simulation programs) consists of an infinite loop with various state labels (e.g., OPERATE, HOLD, RESET, and TERMINATE). The Supervisor routine *"RTMODE"* monitors the current program state and transitions the program to the appropriate state label within the loop. Refer to Appendix C for a high level S-61 program flow chart.

Before any I/O is performed, within the infinite loop, the application program must exit SRT and enter the Moveable Real-Time (MRT) program state (*RTMRT* call) else a lost-time synchronization error occurs and terminates the execution of the program. After the termination of the real-time session, the Supervisor will deallocate all real-time resources previously scheduled.

Currently, the Sikorsky S-61 helicopter flight simulation program is written in FORTRAN F77 version 7.0, uses the second-order one-pass Adams-Bashforth algorithm for its integration scheme, and runs at 32 Hertz (31.25 msec)[6] on the CONVEX C3840 supercomputer system.

The following subsections contain: COMMON block descriptions, software interface arrays, external subroutine descriptions, and a FORTRAN listing of the S-61 real-time application main program followed by a detailed description of its subroutines.

---

[5] *RESADD* call may also be used to give the user an alternative location for recovery within the real-time program.

[6] Currently, any simulation application program running on the ARTSS may have a frame time which is a multiple of 125 microseconds (.125 milliseconds).

COMMON BLOCK DESCRIPTIONS

This section describes the user COMMON blocks and software interface COMMON blocks (see Appendix D for COMMON Block FORTRAN listing).

USER COMMON BLOCKS

The following user defined COMMON blocks are used within the S-61 real-time application program for global data interaction.

User COMMON Blocks	FORTRAN Variable	Description
/ACCEL/	AX	a_x
	AY	a_y
	AZ	a_z
/ADV C/	OMEG	Ω
	WIM	W_{IM}
	VT	V_T
	QV	$\bar{q}_V$
	QL	$\bar{q}_L$
	ASPD	A_{SPD}
/ADV P/	ADVP1	ADV_{P1}
	ADVP2	ADV_{P2}
/CNTL C/	AOSC	A_{OSC}
	A1SC	A_{1SC}
	B1SC	B_{1SC}
	THTRC	θ_{TRC}
/CNTL P/	AOSC0	Control constant in A_{OSC}
	A1SC0	Control constant in A_{1SC}
	B1SC0	Control constant in B_{1SC}
	THTRC0	Control constant in θ_{TRC}
	XAOSG	Control gains in XAOS
	YCSG	Control gains in YCS
	XCSG	Control gains in XCS
	XTRG	Control gains in XTR

User COMMON Blocks	FORTRAN Variable	Description
/CNTSAS/	A1SCS	A_{1SCS}
	B1SCS	B_{1SCS}
	THTRCS	θ_{TRCS}
/COEF/	CT	C_T
	CP	C_P
	CQ	C_Q
	MU	μ
	MTIP	M_{TIP}
	VS	V_S
/CONTRL/	AOS	A_{OS}
	A1S	A_{1S}
	B1S	B_{1S}
	THTR	θ_{TR}
	OMEGD	Ω_D
/CTABLE/		Constant variables table
/C TRIM/		Matrix arrays used to calculate trim values
/DEFLECT/		Cockpit stick deflections, control trim and scaled values
/DIRCOS/		Directional Cosines
/EDERINT/		Equivalenced derivatives
/EINTEG/		Program control variables
/ELOGIC/		Equivalenced control logic array
/ENG C/	QMAX	Q_{MAX}
	QE	Q_E
	SHP	S_{HP}
	GRWT	G_{RWT}
	OMEGDT	$\dot{\Omega}$
	QED	Q_{ED}
/ENG P/	IROT	I_{ROT}
	CKE1	K_{E1}

User COMMON Blocks	FORTRAN Variable	Description
/EUL CS/	COSFI	$\cos(\phi)$
	SINFI	$\sin(\phi)$
	COSTH	$\cos(\theta)$
/EUL CS/	SINTH	$\sin(\theta)$
	SINSI	$\sin(\psi)$
	COSSI	$\cos(\psi)$
/EULER/		Euler angles
/FCNNAME/	CLF1	$C_{lF1} = f(\beta_F)$
	CNF1	$C_{nF1} = f(\beta_F)$
	CX1	$C_{X1} = f(\beta_F)$
	CMF1	$C_{mF1} = f(\alpha_F)$
	CX2	$C_{X2} = f(\alpha_F)$
	CYTR	$C_{YTR} = f(u, V_{TR}, \theta_{TR})$
	CYF	$C_{YF} = f(\beta_F)$
	CZF	$C_{ZF} = f(\alpha_F)$
	CXW	$C_{XW} = f(\alpha_W)$
	CZW	$C_{ZW} = f(\alpha_W)$
	CYVS	$C_{YVS} = f(\beta_{VS})$
	CZHS	$C_{ZHS} = f(\alpha_{HS})$
	CDSIY	$C_{D\psi Y} = f(\alpha_{\psi Y}, U_{T\psi Y})$
	CLSIY	$C_{L\psi Y} = f(\alpha_{\psi Y}, U_{T\psi Y})$
	DW	$D_W = f(u)$
	GEV	$G_{EV} = f(u)$
	GEH	$G_{EH} = f(h)$
/F N M C/	XA	X_A
	YA	Y_A
	ZA	Z_A
	LA	L_A
	MA	M_A
	NA	N_A
/F N M P/	DX	d_X
	DXHS	d_{XHS}
	DXTR	d_{XTR}
	DXVS	d_{XVS}
	DZ	d_Z

User COMMON Blocks	FORTRAN Variable	Description
/F N M P/	DZTR	d_{ZTR}
	DZVS	d_{ZVS}
	DXW	d_{XW}
	DZW	d_{ZW}
	IR	I_R
/FOCOM/		First order transfer function variables
/FTRMPLS/		Trim parameters
/FUNCS/	F001(28)	Array of data values for C_{mF1}
	F002(28)	Array of data values for C_{X2}
	F003(14)	Array of data values for C_{ZF}
	F004(14)	Array of data values for C_{lF1}
	F005(14)	Array of data values for C_{nF1}
	F006(28)	Array of data values for C_{X1}
	F007(14)	Array of data values for C_{YF}
	F008(28)	Array of data values for C_{XW}
	F009(14)	Array of data values for C_{ZW}
	F0010(28)	Array of data values for C_{YVS}
	F0011(35)	Array of data values for C_{ZHS}
	F0013(07)	Array of data values for D_W
	F0014(07)	Array of data values for G_{EV}
	F0015(07)	Array of data values for G_{EH}
	F0016(07)	Array of data values for $C_{D\psi Y}$
	F0017(35,10)	Array of data values for $C_{L\psi Y}$
	F0018(35,10)	Array of data values for C_{YTR}
	D001(9)-D011(9)	Array containing the generated function values and function parameters for F001-F011
	D013(9)-D015(9)	Array containing the generated function values and function parameters for F013-F015
	D016(18)-D017(18)	Array containing the generated function values and function parameters for F016-F017
	D018(27)	Array containing the generated function values and function parameters for F018
/FUS C/	XF	X_F
	YF	Y_F
	ZF	Z_F
	LF	L_F
	MF	M_F

User COMMON Blocks	FORTRAN Variable	Description
/FUS C/	NF	N_F
	XW	X_W
	WIWM	W_{IWM}
	ZW	Z_W
	ALFF	α_F
	BETF	β_F
/FUS C/	ALFW	α_W
/FUS P/	CLF2	C_{lF2}
	CMF2	C_{mF2}
	CNF2	C_{nF2}
	CKRF	$K_{R/F}$
/FUS P/	CKWIWM	K_{WIWM}
	IW	I_W
	VLF	V_{LF}
	VMF	V_{mF}
	VNF	V_{nF}
	SXF1	S_{XF1}
	SXF2	S_{XF2}
	SYF	S_{YF}
	SZF	S_{ZF}
	SXW	S_{XW}
	SZW	S_{ZW}
/GPARAM/	DELT	Δt
	RHO	ρ
	G	g
	PIE	π
	DEG2RAD	Degrees per radian conversion factor
	RAD2DEG	Radians per degree conversion factor
/MROT C/	FXR	F_{XR}
	FYR	F_{YR}
	LMR	L_{MR}
/MROT C/	LRH	L_{RH}
	MRH	M_{RH}
	QMR	Q_{MR}
	AOSS	α_{OSS}
	ALFSIY	$\alpha_{\psi Y}$

User COMMON Blocks	FORTRAN Variable	Description
/MROT C/	UTSIY	$U_{T\psi Y}$
	BETSI	β_{ψ}
	.	.
	BDTSI	β_{ψ}
	A1SS	α_{1SS}
	.	.
	AD1SS	α_{1SS}
	B1SS	β_{1SS}
	.	.
	BD1SS	β_{1SS}
/RCOEFF/	CKL(20)	K_L
	CKD(20)	K_D
	CKQ(20)	K_Q
	CKDL(20)	K_{DL}
	CKQL(20)	K_{QL}
	CKM(20)	K_M
/RCOEFF/	Y(20)	Y
	WF(20)	W_F
/ROTCON/	YPE	Y_{PE}
	YTW	Y_{TW}
	WIF	W_{IF}
	BOVN	Ratio of N_B to N_{AZ}
/ROT P/	TWIST	Blade twist
	FMRS	F_{MRS}
	CKAO	K_{AO}
	CKAB1	K_{AB1}
	CKAB2	K_{AB2}
	CKAB3	K_{AB3}
	NRAD	N_{RAD}
	NAZ	N_{AZ}
	NB	N_B
/ROT P/	RB	R_B
	EFH	e
/SAS2COM/		SAS components variables

User COMMON Blocks	FORTRAN Variable	Description
/SASINPT/		SAS input variables
/SCALES/		Cockpit scale factors
/SCPSIR/	SNSIR	$SIN(\psi_R)$
	COSSIR	$COS(\psi_R)$
/SHIP P/	IXX	I_{XX}
	IYY	I_{YY}
	IZZ	I_{ZZ}
/SHIP P/	MASS	m
/STICKS/	XAOS	X_{AOS}
	YCS	Y_{CS}
	XCS	X_{CS}
	XTR	X_{TR}
/SUMMS/		Summation of moments
/SWIND/		Wind components
/TAIL C/	ALFHS	α_{HS}
	BTVS	β_{VS}
	ZHS	Z_{HS}
	YVS	Y_{HS}
	VTR	V_{TR}
	WHS	W_{HS}
	YTR	Y_{TR}
	DELE	δ_E
/TAIL P/	CKTR1	K_{TR1}
	CKTR2	K_{TR2}
	IHS	I_{HS}
	IVS	I_{VS}
	SZHS	S_{ZHS}
	SYVS	S_{YHS}
/TRIMAT/	XMAT(11,12)	A (NEQN x NEQP1) matrix of independent trim variables
	EMAT(12,13)	A (NEQP1 x NEQP2) matrix of functional evaluations

User COMMON Blocks	FORTRAN Variable	Description
/TRIMAT/	TRSUM(12)	A NEQP1 dimensional vector of error sums
	NEQN	Number of equations to be trimmed
	NEQP1	NEQN + 1
	NEQP2	NEQN + 2
	XMATG(11)	Increments added to XTRIM to start the iteration
/WIND/		Wind direction variables

SOFTWARE INTERFACE COMMON BLOCKS

The following COMMON blocks are used within the S-61 real-time application program to communicate with the Supervisor, linked libraries, as well as with the peripheral hardware (see Appendix D for COMMON Block FORTRAN listing).

Software Interface COMMON Blocks	Description
CFSS	System defined console Function Sense Switches (FSS) variables
CGICOMM	User defined CGI interface variables
CGIZCOM	CGI library defined CGI interface variables
CLITE	System defined RTSC indicator light variables
CMCP	System defined RTSC Mode Control Panel (MCP) switch variables
CMIDI	System sound simulator interface variables
CONSOLE	System defined console I/O variables
CRDCOMM	CRDS library defined CRDS interface variables
C RECDR	User defined strip chart minimum and maximum widths setup variables
DISPLAY	System defined turret labels variables
DSCRETE	User defined packed discrete array variables
FILE1	User defined real-time disk array variables
INTCOMM	System defined integration variables (used by "IGRATE6" integration library subroutine)
LMODE	System defined latching MCP variables (used for batch mode, transparent to user in real-time mode)
MASK	System defined bit masking variables
REALTIM	User defined cockpit I/O variables (used by RTBUFDF and RTMODE)
RTBUFS	System defined Supervisor scratch buffer variables
RUNWAY	User defined runway parameter variables
SKEDS	User defined schedule configuration variables
SSWTCHS	User defined hardware configuration array variables
USERINFO	System defined Supervisor status variables
WCOM	User defined equivalenced variables (used to the linked washout utility library for motion legs)

SOFTWARE INTERFACE ARRAYS

This section describes various arrays used by the S-61 real-time application program to interface with the hardware configuration established by the Supervisor.

Analog-to-Digital Converter

Analog-to-Digital Converter (ADC) is an input array from the VMS cockpit. ADC is declared as a 15 element array of 64-bit REALs initialized to zero. The ADC array contains converted analog-to-digital signals from the VMS cockpit. ADC locations are determined by the associated RTBUFDF call.

Example: Call RTBUFDF ('vmsadc1', ADC, 15, STATMAD)

The Supervisor then scales the 120 bits (15 bytes) into a 16-bit representation in order to receive the data from the real-time I/O system. Refer to 'REALTIM' COMMON block within Appendix D.

The VMS site provides up to 30 possible ADCs of which the S-61 real-time application program utilizes the following:

ADC Element	Description
1	Pitch position
2	Roll position
3	Rudder position
13	Tail rotor pedal trim adjustment input
14	Pedal force
15	Collective position

Digital-to-Analog Converter

Digital-to-Analog Converter (DAC) is an output array to the VMS cockpit. DAC is declared as a 60 element array of 64-bit REALs initialized to zero. The DAC array contains converted digital-to-analog signals from the VMS cockpit. DAC locations are determined by the associated RTBUFDF call.

Example: Call RTBUFDF ('vmsdac1', DAC, 60, STATMDA)

The Supervisor then scales the 480 bits (60 bytes) into a 16-bit representation in order to send the data to the real-time I/O system. Refer to 'REALTIM' COMMON block within Appendix D.
The VMS site provides up to 78 possible DACs of which the S-61 real-time application program utilizes the following:

DAC Element	Description
4	Sideslip angle
14	Rate of turn (for right side)
18	Rate of turn (for left side)
19	Ball angle (for both sides)
20	Indicated Airspeed
25	Course altitude
26	Fine altitude
27	Fine altitude
28	Vertical speed
31	Engine RPM
32	Percent Power
36	Audio
48	Rudder trim Indicator
55	Motion leg #1
56	Motion leg #2
57	Motion leg #3
58	Motion leg #4
59	Motion leg #5
60	Motion leg #6

Synchro

Synchro (SYNCRO) is a digital-to-synchro converter output array to the VMS cockpit. SYNCRO is declared as a 12 element array of 64-bit REALs initialized to zero. The SYNCRO array contains angle values that turn the instrument indicators and/or value settings in the VMS cockpit to the same relative position as existing or established by the simulation program calculation. SYNCRO locations are determined by the associated RTBUFDF call.

Example: Call RTBUFDF ('vmsdsc1', SYNCRO, 12, STATMSO)

The Supervisor then scales the 96 bits (12 bytes) into 16-bit representation in order to send the data to the real-time I/O system. Refer to 'REALTIM' COMMON block within Appendix D.

The VMS site provides up to 12 possible SYNCROs of which the S-61 real-time application program utilizes the following:

SYNCRO Element	Description
4	Pitch angle
5	Bank (roll) angle
9	Indicated Airspeed

Input Discretes

Input Discretes (IDIS) is a logical discrete input array. IDIS is declared as a four element array of 64-bit integers initialized to false. The first element contains the converted ON/OFF signals as 1 or 0 values, respectively, received from the VMS site; the other remaining elements are left for future expansion. IDIS contains packed discrete bits specified by the associated RTBUFDF call.

Example: Call RTBUFDF ('vmsdiz1', IDIS, 4, STATMDO)

This RTBUFDF call tells the Supervisor to get (unpack) 32 bits (4 bytes) from the real-time highway. Refer to 'DSCRETE' COMMON block within Appendix D.

The VMS site provides up to 32 possible input discretes of which the S-61 real-time application program utilizes the following:

IDIS(1) Bit	Description
1	Hold
2	Hold Push Button (on central console)
10	Heading Hold
15	Longitudinal Trim Aft (Pitch Up)
16	Longitudinal Trim Forward (Pitch Down)
17	Lateral Trim Left (Roll Left)
18	Lateral Trim Right (Roll Right)
30	Control Loading Status (True if shutdown occurs)

Output Discretes

Output Discretes (ODIS) is a logical discrete output array. ODIS is declared as a three element array of 64-bit integers initialized to false. The first element contains the converted ON/OFF signals as 1 or 0 values, respectively, sent to the VMS site; the other remaining elements are left for future expansion. ODIS contains packed discrete bits specified by the associated RTBUFDF call.

Example: Call RTBUFDF ('vmsdoz1', ODIS, 3, STATMDO)

This RTBUFDF call tells the Supervisor to send (pack) 24 bits (3 bytes) to the real-time highway. Refer to 'DSCRETE' COMMON block within Appendix D.
The VMS site provides up to 30 possible output discretes of which the S-61 real-time application program utilizes the following:

ODIS(1) Bit	Description
1	OPERATE
2	HOLD
3	RESET

ODIS(1) Bit	Description
7	Thrust Trim Increase
8	Thrust Trim Decrease
9	Longitudinal Trim Aft (Pitch Up)
10	Longitudinal Trim Forward (Pitch Down)
11	Lateral Trim Left (Roll Left)
12	Lateral Trim Right (Roll Right)
13	Rudder Trim Left
14	Rudder Trim Right
17	Excess Collective
23	Heading Hold

Function Sense Switches

Function Sense Switches (FSS) is a logical discrete array input from the console touch screen. FSS is declared as a 24 element array of type logical initialized to false. Refer to 'CFSS' COMMON block within Appendix D (see figure 6).

There are 64 FSS available of which S-61 real-time application program utilizes the following:

FSS Element	Description
1	Dynamic Check - Doublet inputs on specified controls
2	Gusts
3	Dispose To Main Printer
4	Winds
5	Trim
7	Bias Controls
8	Statics - ADC Inputs
9	S-61 SAS Model
10	COBRA SAS Model
13	Cockpit Control
14	Turret
15	Stop CRD Data
16	Send CGI Data
17-23	CGI Setting Conditions (not shown on figure 6)
24	Sound Dump Buffer (not shown on figure 6)

Mode Control Panel Switches

Mode Control Panel (MCP) switches is a logical discrete array input from the console touch screen. MCP is declared as a 10 element array of type logical initialized to false. Refer to 'CMCP' COMMON block within Appendix D (see figure 6).

MCP Element	Description	
1	OPERATE	Integration of equations of motion
2	HOLD	Stops integration of equations of motion and holds all variables at present values
3	RESET	Initialization of variables (condition for t=0)
4	RESTART	Restart the RTSC (reloads SMM)
5	TERM	Terminates execution of program
6	ERASE1	Erase Real-time disk 1
7	ERASE2	Erase Real-time disk 2
8	PANEL REFRESH	Refresh console touch screen
9	PRINT	Prints output of previously stored variables
10	READ	Reads in the Turret Programmable Display Pushbutton (PDP) pixel file

White Lights

White Lights (WLITE) is a logical discrete array of program indicators output to the console touch screen. WLITE is declared as a 48 element array of type logical initialized to false. The WLITE array is defined in the RTSC touch screen program as white; therefore, an element will turn on a white light whenever set to true in the corresponding position on the screen. Refer to 'CLITE' COMMON block within Appendix D (see figure 6).

WLITE Element	Description
1	Trim Good
5	SAS ON
10	Heading Hold
13	ALFF beyond limit
14	BETF beyond limit
15	ALFW beyond limit
16	ALFHS beyond limit
17	BTVS beyond limit
18	UTSIY beyond limit
19	ALFSIY beyond limit
20	CYTR Error
21	Collective
22	Longitudinal Cyclic
23	Lateral Cyclic
24	Pedal
30	BEEP!
36	MB Reset
38	No FLDIR
41	Trim Bad
42	AOS Limit
43	A1S Limit

WLITE Element	Description
44	B1S Limit
45	THTR Limit
46	Nerr => True
47	H .LE. 6.5
48	Q .GT. Qmax

Red Lights

Red Lights (RLITE) is a logical discrete array of program indicators output to the console touch screen. RLITE is declared as an eight element array of type logical initialized to false. RLITE array elements one through eight are mapped (by an equivalence statement) with the corresponding WLITE array elements 41 through 48. RLITE array element is set true when its specific conditions are met (see description below); this array element value then overrides the corresponding WLITE array element value. The RLITE array is defined in the RTSC touch screen program as red; therefore, an element will turn on a red light whenever set to true in the corresponding position on the screen. Refer to 'CLITE' COMMON block within Appendix D (see figure 6).

RLITE Element	Description
1	Unsuccessful trim
2	AOS beyond set limits
3	A1S beyond set limits
4	B1S beyond set limits
5	THTR beyond set limits
6	PHI or THETA has become too large
7	Negative altitude
8	Engine power greater than maximum

EXTERNAL SUBROUTINE DESCRIPTIONS

This section describes the linked external subroutines from the Supervisor libraries and supplementary libraries used by the S-61 real-time application program.

REAL-TIME SUPERVISOR SUBROUTINE DESCRIPTIONS

The Real-Time Supervisor consists of a group of subprograms (libraries) which are loaded with each simulation application. The Supervisor provides the interface between the application program and the operating system, and coordinates I/O to and from the simulation hardware. The following subroutines are loaded by S-61 real-time application program.

CALLIGRAPHIC RASTER DISPLAY SYSTEM (CRDS) SUBROUTINES

The following subroutines are linked from the "usr/local/lib/libcrds.a" Supervisor CRDS utility library. This subroutines provide software control of the CRDS from a real-time simulation program.

Subroutines	Description
CRDBLK	CRDS data blocking
CRDCHNG	CRDS channel change
CRDINIT	CRDS initialization
CRDXFR	CRDS data transfer
FLP2CRD	Convert to CRDS floating point format
INT2CRD	Convert to CRDS integer format

COMPUTER-GENERATED IMAGE (CGI) SYSTEM SUBROUTINES

The following subroutines are linked from the "/usr/local/lib/libcgi.a" Supervisor CGI utility library which provide software control of the CGI system from a real-time simulation program.

Subroutines	Description
CGICLD	Cloud layer and scudding control
CGICOLD	Collision Detection
CGIDCS	Dynamic Coordinate System (DCS) (eyepoint) positioning
CGIERRP	Host command error processing and data transfer dump
CGIEYEO	Eyepoint offset from DCS 0
CGIFLSH	Lightning flash control
CGIFOG	Fog layer control
CGIHAT	DCS height above terrain initialization
CGIHATR	DCS height above terrain data retrieval

Subroutines	Description
CGIINIT	CGI runway and data transfer initialization
CGILOBE	Aircraft landing light pattern specification
CGILUM	Sun, fog, and polygon illumination
CGIMLGC	Airport multiple light group definition
CGIMSEQ	Model sequencing initialization
CGIMSTP	Model sequencing stop command
CGIRTN	CGI return data block control
CGISLGC	Single light group control
CGISTAT	CGI return block information
CGISUN	Sun position specification
CGITXMO	Texture motion control
CGIVIS	DCS visibility control
CGIXFR	Transmits data to CGI

REAL-TIME SYSTEM CONSOLE (RTSC) SUBROUTINES

The following subroutines are linked from the "/usr/local/lib/librts.a" Supervisor RTSC utility library. These subroutines execute, monitor, and control a real-time simulation program operating within the ARTSS.

Subroutines	Description
ATERM	Ends the sequence of a terminate cycle and puts the user back into synchronized real-time (entry point in RTMODE)
FUNC1	One dimension function table lookup
FUNC2	Two dimension function table lookup
FUNC3	Three dimension function table lookup
ICONV	Packs a sequence of string and integer values into an integer Hollerith word
IGRATE6	Adams-Bashforth second-order one-pass integration scheme
PACKC	Packs mode, function, and light data into RAM2 buffer space
RNNDH1	Generates normally distributed random numbers
RNNDH2	Same function as RNNDH1; allows starting the random number generator with a different seed
RNNDH3	Same function as RNNDH1; allows starting the random number generator with a different seed
RTMODE	Real-time mode control
SETUP	Packs the Prolog values into the RAM2 buffer space for the Engineer Control Console (ECC) also known as Turret station
SETUPIC	Initializes TMASK, FMASK, and looping variables used in PACKC and UNPACKC
SETUP0	Attachs a termination bit to the last RAM word packed
UNPACKC	Unpacks mode and function switch data from RAM0 buffer

REAL-TIME SYSTEM MAIN SUBROUTINES

The following subroutines are linked from the "/usr/local/lib/libsuper.a" Supervisor main utility library. These subroutines provide the interface between the application program and the operating system, and coordinates I/O to and from the simulation hardware.

Subroutines	Description
FILEARR	Establishes memory and variable table for the real-time data recording
FILEVAR	Puts addresses in variable table real-time data recording
LOSTIME	Enables lost-time synch recovery capability
NEWVAR	Clears table created by FILEVAR
RECADD	Defines an error recovery address
RESADD	Defines a resumption address for RTGO
RTBUFDF	Maps program variables to hardware I/O
RTCLASS	Determines which Supervisor is being used
RTCOMDF	Initializes the Supervisor and establishes the real-time highway communication buffer
RTCYCLE	Signals the Serial Highway Interface Processor (SHIP) to do end of frame processing
RTMRT	Enters MRT program state
RTNETOT	Output of asynchrous data
RTOUT	Routes a file to a printer
RTREAD	Reads one set of values from the real-time disk
RTSRT	Enters SRT program state
RTWRITE	Writes one set of program variables to a simulation data recording file
REWRTF	Rewinds real-time data recording

SUPPLEMENTARY SUBROUTINE DESCRIPTIONS

The following subroutines are linked either from "/usr/lib/libF77.a", "~geh/Srclibs/scrclib5", "~mart/S61MB/S61bin" or "~swc/Washlib/washhlib" utility libraries. These subroutines perform the described functions.

Subroutines	Description
BATCH	Controls program flow without console input
BULKDAT	Initializes data input values
DABS	Double precision absolute function
DBLE	Double precision function
DEADBND	Applies a symmetric deadband about zero to the input variable
FOLAG	First-order lag
LJUST	Left justifies character string
SNGL	Single precision function
SSWTCH	Reads environment variables that simulate computer sense switches
WCNTRL	Controls the washout subroutine calls
WINIT	Loads the initial values into the washout parameter arrays

S-61 REAL-TIME PROGRAM

This section contains the FORTRAN listing of the S-61 real-time application main program (*s61mb.f*) followed by a detailed description of the 46 subroutines that comprise the body of the simulation application.

Refer to Appendix C for a high level S-61 program flow chart, Appendix D for the FORTRAN listing of the S-61 COMMON blocks of global variables, and Appendix E for a FORTRAN listing of "BULKDAT" subroutine which initializes variables within COMMON blocks listed in Appendix D.

S-61 REAL-TIME APPLICATION MAIN PROGRAM FORTRAN LISTING *(S61MB.F)*

```
      PROGRAM S61MB
C*
C*************************************************************
C*
C*     VMS REAL-TIME SIMULATION
C*
C*     SIMULATION OF A SIKORSKY S-61 HELICOPTOR FOR
C*     THE 6-DOF VISUAL MOTION BASE SIMULATOR
C*
C*************************************************************
C*
*call accel.com
*call advc.com
*call advp.com
*call cgicomm.com
*call cgizcom.com
*call cmidi.com
*call cntic.com
*call cntip.com
*call cntsas.com
*call coef.com
*call console.com
*call contrl.com
*call crdcomm.com
*call crecdr.com
*call ctable.com
*call ctrim.com
*call deflect.com
*call dircos.com
*call display.com
*call decrete.com
*call eintag.com
*call elogic.com
*call engc.com
*call engp.com
*call eulcs.com
*call euler.com
*call fcnname.com
*call file1.com
*call fnmc.com
*call fnmp.com
*call focom.com
*call ftrmpls.com
*call funcs.com
*call fusc.com
*call fusp.com
*call gparam.com
*call intcomm.com
*call ederint.com
*call lmode.com
*call mask.com
*call mrotc.com
*call realtim.com
*call rcoeff.com
```

```
*call rotp.com
*call rtbufs.com
*call runway.com
*call sas2com.com
*call sasinpt.com
*call scales.com
*call scpsir.com
*call shipp.com
*call skeds.com
*call swtchs.com
*call sticks.com
*call summs.com
*call swind.com
*call talic.com
*call talip.com
*call trimat.com
*call userinfo.com
*call wcom.com
*call wind.com
      LOGICAL MMM
      LOGICAL RUNINC
*call climit.com
C*********************************** ******
C
C*   FILE IDENTIFICATION:
C*
C*   UNIT #     FILE NAME        DESCRIPTION
C*   ======     =========        ===========
C*
C*     3      ->  config              -> HARDWARE CONFIGURATION DATA
C*     6      ->  standard output     -> HARDWARE CONFIGURATION DATA
C*   100      ->  BATCHOUT            -> BATCH RUN OUTPUT
C*   110      ->  CHECKCASE           -> (STATIC OR DYNAMIC) CHECK CASE DATA
C*   120      ->  COMBAT              -> DATA FOR COMBAT MONITOR PROGRAM (CMP)
C*   130      ->  OUTPUT              -> DATA FOR CHECK CASE GRAPHS
C*
C*********************************** ******
C*
      OPEN(3, FILE='config')
C
C*   UNPACK THE SENSE SWITCHES
C
      DO 50 ISW = 1, 6
   50   CALL SSWTCH(ISW, ISSWTCH(ISW))
C
C*   RTCLASS - SETS THE BATCH FLAG
C
      CALL RTCLASS( VBATCH )
C
      CALL CONFIG
C
      CALL INIT
C
C* INITIALIZE THE SOUND HARDWARE
C
      IF ( CABON ) CALL ICSOUND
C
C*   SET UP THE RECOVERY AND RESUME ADDRESSES
C
      CALL RESADD(1, 1)
      CALL RECADD(1, 1)
C
C*   IGNORE HIGHWAY DEMANDS
C
      CALL DMDERRS(100)
C
C*   SET UP LOST-TIME SYNC RECOVERY
C
      CALL LOSTIME(0,1)
C
C*   ENTER SYNCRONIZED REAL-TIME (SRT)
C
      CALL RTSRT
C
C*   IF RESUMING PROCESSING, CONTINUE IN THE PROPER MODE
C
      IF ( OPERATE ) GO TO 90006
      IF ( HOLD    ) GO TO 90002
C
```

```
      CALL SETUPIC
      CALL PACKC
      CALL SETUP0
      CALL UNPACKC
C
C*****************************************************************
C
C*  R E S E T   L O O P  [90003]
C
C*****************************************************************
C
90003 CONTINUE
C
      IF ( CABON ) CALL SOUNDHWIC
C
      AUTOHW = .FALSE.
C
      IF ( MCASE .EQ. 1 ) THEN   ! ALIGN W/RUNWAY 26 L (WEST)
C
C*    0 KNOTS/12 FEET INITIAL CONDITIONS
C
       KNOTS  = 1
       H0     = 12.          ! INITIAL EARTH VERTICAL COORDINATE
       PSI0   = 269.925      !  "   FUSELAGE YAW ANGLE, RADIANS
       N0     = 0.001        ! INITIAL EARTH COORDINATE IN NORTH DIRECTION
       E0     = 0.001        !  "   EARTH COORDINATE IN EAST DIRECTION
C
      ELSE IF ( MCASE .EQ. 2 ) THEN
C
C*    0 KNOTS/50 FEET INITIAL CONDITIONS
C
       KNOTS  = 1
       H0     = 50.
       PSI0   = 180.
       N0     = -500.
       E0     = 0.
C
      ELSE IF ( MCASE .EQ. 3 ) THEN
C
C*    80 KNOTS/800 FEET INITIAL CONDITIONS
C
       KNOTS  = 3
       H0     = 800.
       PSI0   = 180.
       N0     = 5000.
       E0     = 0.
C
      ELSE IF ( MCASE .EQ. 4 ) THEN
C
C*    80 KNOTS/272 FEET INITIAL CONDITIONS
C
       KNOTS  = 3
       H0     = 272.
       PSI0   = 210.
       N0     = 4330.
       E0     = 2500.
C
      ELSE IF ( MCASE .EQ. 5 ) THEN
C
C*    80 KNOTS/350 FEET INITIAL CONDITIONS
C
       KNOTS  = 3
       H0     = 350.
       PSI0   = 225.
       N0     = -25837.
       E0     = 9563.
C
      ELSE IF ( MCASE .EQ. 6 ) THEN
C
C*    80 KNOTS/350 FEET INITIAL CONDITIONS
C
       KNOTS  = 3
       H0     = 350.
       PSI0   = 0.
       N0     = -10000.
       E0     = 0.
C
      END IF
C
```

```
      MCASE = 0
C
C*    SELECT PROFILE: KNOTS  -  CONDITION
C*                    ----     ----------  ---------
C*                      1        0 KNOTS - HOVER
C*                      2       33 KNOTS
C*                      3       80 KNOTS - DYNAMIC CHECK CASE
C*                      4      105 KNOTS
C*                      5      120 KNOTS
C*                      6       80 KNOTS - STATIC CHECK CASE
C
      IF ( KNOTS .EQ. 1 ) THEN
C
C*      HOVER INITIAL CONDITIONS
C
        AOS0   = 0.2738429
        A1S0   = -0.031216394
        B1S0   = -0.083409593
        THTR0  = 0.1382389
        PHI0   = -0.019283726
        THETA0 = -0.062152122
C
        IF ( H0 .LE. 100. ) THEN
C
C*        12 FEET HOVER
C
          AOS0   = 0.244563E+00
          A1S0   = -0.331366E-01
          B1S0   = -0.829220E-01
          THTR0  = 0.120902E+00
          PHI0   = -0.155973E-01
          THETA0 = -0.619235E-01
C
        END IF
C
        WIM    = 0.351603E+02
        AOSS   = 0.347271E-01
        A1SS   = 0.829221E-01
        B1SS   = -0.331365E-01
        UKNOTS = 0.
        V0     = 0.
        W0     = 0.
C
      ELSE IF ( KNOTS .EQ. 2 ) THEN
C
C*      33 KNOTS INITIAL CONDITIONS
C
        AOS0   = 0.2462616      ! INITIAL MAIN ROTOR COLLECTIVE PITCH
        A1S0   = -0.024896768   !  "   "   "  LATERAL CYCLIC PITCH
        B1S0   = -0.044236309   !  "   "   "  LONGITUDINAL CYCLIC PITCH
        THTR0  = 0.080909505    !  "  FUSELAGE ROLL ANGLE
        THETA0= -0.073933757    !  "  FUSELAGE PITCH
        PHI0   = -0.014087214
        WIM    = 9.0929126E+00  ! MAIN ROTOR MEAN INFLOW VELOCITY
        AOSS   = 4.4615891E-02
        A1SS   = 6.6313158E-02
        B1SS   = -1.7969689E-03
        UKNOTS= 33.15571344
        V0     = 0.00
        W0     = -4.148259
C
      ELSE IF ( KNOTS .EQ. 3 ) THEN
C
C*      80 KNOTS/800 FEET INITIAL CONDITIONS (DYNAMIC CHECK CASE)
C
        AOS0   = 2.3983840E-01
        A1S0   = -2.0950998E-02
        B1S0   = -1.6629833E-02
        THTR0  = 2.9319165E-02
        THETA0 = -1.0237187E-01
        PHI0   = -1.2824267E-02
        WIM    = 9.0929126E+00
        AOSS   = 4.4615891E-02
        A1SS   = 6.6313158E-02
        B1SS   = -1.7969689E-03
        UKNOTS = 79.928952
        V0     = 0.00
        W0     = -1.3869790E+01
C
```

```
      ELSE IF ( KNOTS .EQ. 4 ) THEN
C
C*    105 KNOTS INITIAL CONDITIONS
C
      AOS0   = 0.2591912
      A1S0   = -0.018925199
      B1S0   = 0.021399861
      THTR0  = 0.031807306
      PHI0   = -0.026224227
      THETA0 = -0.1159836
      WIM    = 9.0929126E+00
      AOSS   = 4.4615891E-02
      A1SS   = 6.6313158E-02
      B1SS   = -1.7969869E-03
      UKNOTS = 105.1509769
      V0     = 0.00
      W0     =-20.68070
C
      ELSE IF ( KNOTS .EQ. 5 ) THEN
C
C*    120 KNOTS INITIAL CONDITIONS
C
      AOS0   = 0.2791526
      A1S0   = -0.021712153
      B1S0   = 0.045706954
      THTR0  = 0.036306344
      PHI0   = -0.034026789
      THETA0 = -0.1291323
      WIM    = 9.0929126E+00
      AOSS   = 4.4615891E-02
      A1SS   = 6.6313158E-02
      B1SS   = -1.7969869E-03
      UKNOTS =120.0
      V0     = 0.00
      W0     =-26.33423
C
      ELSE IF ( KNOTS .EQ. 6 ) THEN
C
C*    80 KNOTS/5000 FEET INITIAL CONDITIONS (STATIC CHECK CASE)
C
      AOS0   = 2.39685400E-01
      A1S0   = -1.66214530E-02
      B1S0   = -1.18307260E-02
      THTR0  = 2.93136550E-02
      PHI0   = -1.70735810E-02
      THETA0 = -9.73077730E-02
      PSI0   = 0.0
      WIM    = 9.09481658E+00
      AOSS   = 4.45834208E-02
      A1SS   = 6.14689069E-02
      B1SS   = 2.52623282E-03
C
      UKNOTS = 79.928952
      V0     = 0.0
      W0     = -1.31800900E+01
      H0     = 5000.0
C
      END IF
C
      U0     = UKNOTS * 1.689
C
      AOSC   = AOS0
      A1SC   = A1S0
      B1SC   = B1S0
      THTRC  = THTR0
C
C*    STEADY WIND INITIALIZATION
C
      PSIWR  = PSIWD0 * DEG2RAD
      VWIND  = VWIND0 * 1.689
C
C*    GUST RMS INTENSITY INITIALIZATION
C
      SIGMAU = SIGMAU0
C
C*    IF FSS( 5) IS ACTIVATED, PERFORM TRIM CIRCUIT ALGORITHM
C
      IF ( FSS( 5)) THEN
        CALL RTMRT
```

```
      CALL HELITRIM
      CALL RTSRT
    END IF
C
    XAOST  = (AOS0  * RAD2DEG - AOSC0 ) / XAOSG
    XCST   = (B1S0  * RAD2DEG - B1SC0 ) / XCSG
    YCST   = (A1S0  * RAD2DEG - A1SC0 ) / YCSG
    XTRT   = (THTR0 * RAD2DEG - THTRC0) / XTRG
C
C*  INITIALIZE QUATERNION VARIABLES
C
    CPSI2  = COS(0.5*PSI)
    SPSI2  = SIN(0.5*PSI)
C
    CTHETA2 = COS(0.5*THETA)
    STHETA2 = SIN(0.5*THETA)
C
    CPHI2  = COS(0.5*PHI)
    SPHI2  = SIN(0.5*PHI)
C
    AB1    = CPSI2*CTHETA2*CPHI2 + SPSI2*STHETA2*SPHI2
    AB2    =-CPSI2*STHETA2*SPHI2 + SPSI2*CTHETA2*CPHI2
    AB3    = CPSI2*STHETA2*CPHI2 + SPSI2*CTHETA2*SPHI2
    AB4    = CPSI2*CTHETA2*SPHI2 - SPSI2*STHETA2*CPHI2
C
C*  SET A FLAG SO THE RUN NUMBER IS AUTOMATICALLY INCREMENTED ON
C*  THE FIRST PASS THROUGH OPERATE
C
    RUNINC = .TRUE.
    AUTOH  = .FALSE.
    MMM    = .FALSE.
C
    NERR   = .FALSE.
    INT    = 0
C
C*  ISCHEMES: 1 = 2ND ORDER RUTTAN KATA
C*            2 = 4TH ORDER RK
C*            3 = 2ND ORDER AM
C*            4 = 4TH ORDER AM
C*            5 = 1ST ORDER ADAMS-BASHFORTH INTEGRATION
C
    ISCHEME = INTSCME
C
    IF ( INTSCME .EQ. 5 ) ISCHEME = 3
C
    DELT   = DT
    DX     = (CG - 200.) / 12.
    GRWT   = GRWT0
    HDOTD  = HDOTD0 / 60.
    LAPLACE = LAPLAC0
    MASS   = GRWT / G
    OMEGD  = OMEG0 / REVTRAD
    OMEG   = OMEGD
    T      = 0.
    VS     = 1100.
C
    P      = P0
    Q      = Q0
    R      = R0
C
    PHI    = PHI0
    THETA  = THETA0
    PSI    = PSI0 * DEG2RAD
    PSIHH  = PSI
C
    U      = U0
    V      = V0
    W      = W0
    N      = N0
    E      = E0
    H      = H0
C
    UW     = 0.
    VW     = 0.
    WW     = 0.
C
C*****************************************************************
C*
C*  HOLD  LOOP   [90002]
```

```
C*
C*************************************************************
C
90002 CONTINUE
C
C*************************************************************
C*
C*  RECYCLE  LOOP [90006]
C*
C*************************************************************
C
90006 CONTINUE
C
   IF ( CABON ) CALL SOUNDHWIC
C
C*  UNPACK THE MODES AND FUNCTION SWITCHES
C
   CALL UNPACKC
C
   OPERATE = MCP( 1)
   HOLD    = MCP( 2)
   RESET   = MCP( 3)
C
C*  TURN TRIM LIGHT OFF
C
   IF (WLITE(1) .OR. RLITE(1)) FSS(5) = .FALSE.
C
   IF ( .NOT.OPERATE ) HOLDFLG = .FALSE.
C
C*  DISCRETES 1 AND 2 FROM THE COCKPIT ARE THE PANIC BUTTONS
C
   IF (ISHFT(ID1,64-41) .LT. 0 .OR. ISHFT(ID1,64-42) .LT. 0)
  *  HOLDFLG = .TRUE.
C
   IF ( HOLDFLG ) THEN
    MCP(3)  = .FALSE.
    LMCP(3) = .FALSE.
    RESET   = .FALSE.
C
    MCP(2)  = .TRUE.
    LMCP(2) = .TRUE.
    HOLD    = .TRUE.
C
    MCP(1)  = .FALSE.
    LMCP(1) = .FALSE.
    OPERATE = .FALSE.
   END IF
C
C*  INITIALIZE CGI AND CRD
C
   CALL CGIINTF
   CALL CRDS61
C
C*  CONVERT VALUES TO DEGREES
C
   PSIDEG  = PSI  * RAD2DEG
   THEDEG  = THETA * RAD2DEG
   PHIDEG  = PHI  * RAD2DEG
   ALFFDEG = ALFF * RAD2DEG
   BETFDEG = BETF * RAD2DEG
C
C*  SEND THE SIGNALS TO THE COCKPIT
C
   CALL DACOUT
C
   IF ( .NOT.OPERATE ) THEN
     CALL PACKC
     TURRETON = .FALSE.  .OR.  FSS(14)
     IF (TURRETON) CALL TURRET
C
C*    CALL HANDLES INPUT & OUTPUT TO THE CONSOLE TURRET
C
     IF (TURRETON) CALL TURET
C
   END IF
C
C*  IF RUNNING IN BATCH, CALL THE BATCH CONTROLLER
C
   IF ( VBATCH .EQ. 0 ) CALL BATCH
```

```
C
      IF ( H .LE. 6.5 ) AUTOH = .TRUE.
C
C*    IF FSS(1) IS ON, PERFORM A DYNAMIC CHECK; APPLY A STEP
C*    OR DOUBLET TO THE SPECIFIED PILOT INPUT OR CONTROL SURFACE
C
      IF ( FSS( 1) ) CALL DYNCK
C
C*    SEND DATA TO STRIP CHART RECORDERS FOR DYNAMIC CHECK RUN
C
      IF ( CHARTON ) CALL RECDRS
C
C*    IF FSS(13) IS ON, GET THE INPUTS FROM THE COCKPIT  (COCKPIT CONTROL)
C
      IF ( FSS(13) .AND. (INT .LE. 1) ) CALL ADCIN
C
      XAOS   = (AOSC  * RAD2DEG - AOSC0 ) / XAOSG
      XCS    = (B1SC  * RAD2DEG - B1SC0 ) / XCSG
      YCS    = (A1SC  * RAD2DEG - A1SC0 ) / YCSG
      XTR    = (THTRC * RAD2DEG - THTRC0) / XTRG
C
      THETAPR = THETA - THETA0
      PHIPR   = PHI   - PHI0
      B1SCPR  = B1SC  - B1S0
      A1SCPR  = A1SC  - A1S0
C
      IF ( INT .LE. 1 ) CALL SAS
C
C*    SET CONTROL AUGMENTATION TO ZERO IF THE SAS IS NOT SELECTED
C
      IF ( .NOT.(FSS( 9) .OR. FSS(10)) ) THEN
        AOSCS  = 0.
        A1SCS  = 0.
        B1SCS  = 0.
        THTRCS = 0.
      END IF
C
C*    CALCULATE BLADE ANGLE CONTROLS
C
      AOS   = CLIMIT( AOSC        , AOSLIM (1), AOSLIM (2) )
      A1S   = CLIMIT( A1SC  + A1SCS , A1SLIM (1), A1SLIM (2) )
      B1S   = CLIMIT( B1SC  + B1SCS , B1SLIM (1), B1SLIM (2) )
      THTR  = CLIMIT( THTRC + THTRCS, THTRLIM(1), THTRLIM(2) )
C
C*    RED LIGHTS 2, 3, 4, OR 5 ON RESPECTIVELY INDICATES AOS, A1S,
C*    B1S, OR THTR IS BEYOND SET LIMITS
C
      RLITE( 2) = .FALSE.
      RLITE( 3) = .FALSE.
      RLITE( 4) = .FALSE.
      RLITE( 5) = .FALSE.
C
      IF ( AOS  .EQ. AOSLIM (1) .OR. AOS  .EQ. AOSLIM (2) )
     *   RLITE( 2) = .TRUE.
      IF ( A1S  .EQ. A1SLIM (1) .OR. A1S  .EQ. A1SLIM (2) )
     *   RLITE( 3) = .TRUE.
      IF ( B1S  .EQ. B1SLIM (1) .OR. B1S  .EQ. B1SLIM (2) )
     *   RLITE( 4) = .TRUE.
      IF ( THTR .EQ. THTRLIM(1) .OR. THTR .EQ. THTRLIM(2) )
     *   RLITE( 5) = .TRUE.
C
C*    CALL THE GUST AND WIND MODELS IF DESIRED
C
      UGUST  = 0.
      VGUST  = 0.
      WGUST  = 0.
C
      IF ( FSS( 2) .AND. (INT .LE. 1) ) CALL GUST
C
      CALL AEROVAR
C
      UW     = 0.
      VW     = 0.
      WW     = 0.
C
      IF ( FSS( 4) .AND. (INT .LE. 1) ) CALL WINDS
C
      CALL FUS
      CALL TAIL
```

```
      CALL MAINROT
      CALL ENGINE
      CALL F AND M
      CALL BODYDER
C
      IF ( NERR ) AUTOH = .TRUE.
C
      IF ( AUTOH ) MMM = .TRUE.
C
      CALL EARTH M
C
C*   FSS(12) FOR S.AND.L. REAL WORLD FLIGHT DIRECTOR, N/A
C
C     IF ( FSS(12) ) CALL FLDIR
C
      LITE(36) = .FALSE.
      IF ( FSS(12) ) LITE(36) = .TRUE.
C
C*   COMMUNICATE WITH THE MOTION BASE DRIVE ROUTINES
C
      CALL MOTION
C
      IF ( H .LT. 7.0 ) THEN
C
       IF ( HDOT .LE. 0.0 ) AUTOHW = .TRUE.
C
       IIII0   = IIII0 + 1
C
       IF ( (IIII0 .GE. 320) .AND. (HDOT .GT. 0.) ) THEN
        AUTOHW  = .FALSE.
       ELSE
        IF ( VERTVEL ) HDOT = 0.
       END IF
C
      ELSE
C
       IIII0   = 0
C
      END IF
C
      IF ( TAXI .AND. (H .LE. 7.0) .AND. (HDOT .LT. 0.) ) HDOT = 0.
C
      RLITE( 6) = .FALSE.
      RLITE( 7) = .FALSE.
C
      IF ( AUTOH ) THEN
C
C*     RED LIGHT 6 ON: PHI OR THETA HAS BECOME TOO LARGE
C*     RED LIGHT 7 ON: NEGATIVE ALTITUDE
C
       IF ( NERR )     RLITE( 6) = .TRUE.
       IF ( H .LE. 6.5 ) RLITE( 7) = .TRUE.
C
      END IF
C
C*   BRANCH ACCORDING TO THE MODE SELECTED
C
C*   CALL RTMODE(OPERATE,HOLD,RESET,TERMINATE,PRINT,READ)
C
      CALL RTMODE(*90001, *90002, *90003, *90004, *90014, *90015)
C
C*****************************************************************
C*
C*   O P E R A T E   L O O P [90001]
C*
C*****************************************************************
C
90001 CONTINUE
C
      IF ( CABON ) CALL SOUNDHWIC
C
      MBREADY = .FALSE.
C
C*   INCREMENT THE RUN NUMBER AND SET THE COUNTER FOR RECORDING ON
C*   THE REAL-TIME FILE
C
      IF ( RUNINC ) THEN
       NTCNT  = NT    - 1
C
```

```
C*    REWIND THE REAL-TIME FILE 1
C
      CALL REWRTF(1)
      ITER = 1
      RUNINC = .FALSE.
     END IF
C
C*   RECORD ON REAL-TIME FILE
C
     IF ( IPRINT ) THEN
C
      NTCNT  = NTCNT  + 1
C
      IF ( NTCNT .EQ. NT ) THEN
       NTCNT  = 0
       NPTS   = NPTS   + 1
       CALL RTWRITE(1)
      END IF
C
     END IF
C
     IF ( (INT .GT. 1) .AND. (.NOT. AUTOH) ) GO TO 90005
C
     CALL QUAT
     CALL IGRATES
C
     CALL PACKC
     CALL SETUP0
C
     IF ( MMM .OR. AUTOH ) THEN
      CALL RTCYCLE
      GO TO 90006
     END IF
C
90005 CONTINUE
C
C*****************************************
C*
C*   INTEGRATION SEGMENT      [90005]
C*
C*****************************************
C
C*   IF INTSCME = 5, USE ADAMS-BASHFORTH INTEGRATION
C
     IF ( INTSCME .EQ. 5 ) INT = 1
C
C*   INTEGRATION FLOW CONTROL
C
     IF ( INT .LE. 1 ) THEN
      CALL RTCYCLE
     END IF
C
     GO TO 90006
C
90014 CONTINUE
C
C*************************************************************
C*
C*   P R I N T    L O O P     [90014]
C*
C*************************************************************
C
     CALL PRINTSUB
     GO TO 90006
C
90015 CONTINUE
C
C*************************************************************
C*
C*   R E A D     L O O P     [90015]
C*
C*************************************************************
C
     CALL RTMRT
C
C*   DETERMINE IF PICTURE DATA IS TO BE READ IN (picture.dat)
C
     IF (READPIC) CALL READPIXEL
     CALL RTSRT
```

```
      CALL RTSRT
      MCP(10) = .FALSE.
      GOTO 90006
C
90004 CONTINUE
C
C*****************************************************************
C*
C*  TERMINATE LOOP    [90004]
C*
C*****************************************************************
C
C*  TURN OFF ALL TONES WHEN LEAVING REAL-TIME
C
      IF ( CABON ) CALL MIDIEND
C
C
C*   CALL ATERM(OPERATE,HOLD,RESET,TERMINATE,PRINT,READ)
C
      CALL ATERM(*90001,*90002,*90003,*90004,*90014,*90015)
C*
      END
```

S-61 REAL-TIME SUBROUTINE DESCRIPTIONS

The following contains a detailed description of the 46 S-61 real-time application FORTRAN subroutines.

Subroutines	Description
ADCIN	Scales the pilot inputs from the ADCs and solves the control equations for the helicopter collective and cyclic stick inputs, and the tail rotor pedal inputs. This subroutine allows for two modes of operation: primary control is from the control sticks and pedals in the cockpit; and alternate control is from the turret through the use of potentiometers representing the control sticks and pedals.
AEROVAR	Computes the main rotor inflow velocity, the lateral and vertical dynamic pressures, and the fuselage airspeed using general aerodynamic variable equations.
BATCH	Acts as the batch controller for nonreal-time analysis. This subroutine replaces the system subroutine RTSC. BATCH subroutine will put the program into five iterations of reset, five iterations of hold, then 100 iterations of operate. At the end of the operate cycle, the program will go into one iteration of print and then terminate the execution of the real-time program.
BODYDER	Computes total body motion angular and linear accelerations. This subroutine contains body derivative equations and provides a safety feature to prevent program abort if Phi or Theta become large.
CGIINTF	Performs the executive CGI system driver functions for the S-61 real-time program.
CONFIG	Reads the setup configuration file which is used to schedule necessary facility resourses (e.g., real-time highway, console, and simulator site).
CRDS61	Performs the executive CRDS driver functions for the S-61 real-time program.
DACOUT	Solves equations and logic needed to drive cockpit instruments and motion legs. Provides values to drive the following instruments: altimeter, rate of climb indicator (vertical speed), rate of turn indicator, main rotor rpm indicator, and airspeed indicator.
DYNCK	Performs a dynamic check; applies a step or doublet to the specified pilot input or control surface.

Subroutines	Description
EADI	Performs the EADI display driver functions for sending data to the CRDS.
EARTH M	Computes earth rates (earth orientation equations) by resolution of body axis linear velocities through euler angles.
ENGINE	Computes required engine power (torque), maximum engine power available, and main rotor angular velocity.
EVAL	Sets up a (N+1)X(N+1) matrix of functional evaluations used by the trim circuit algorithm computes (N+1) sums; where each sum is the sum of the trim equations squared for each column.
F AND M	Solves force and moment equations used to calculate the total aerodynamic forces and moments acting on helicopter.
FO	Computes first order transfer function.
FOPQ	Computes constants used in solution of first order transfer function.
FUS	Solves fuselage equations and functions used in calculating the fuselage forces and moments. Also calculates the wing lift, drag force, and downwash velocity.
GUST	Computes gust components along body axes which, if selected, may be added to the body velocities.
HELITRIM	Solves the trim circuit algorithm. If function switch five [FSS(5)] is on, routine iterates control deflections until state variable derivatives are zero.
ICSOUND	Synchronizes the sound hardware by transmitting three zero bytes. The sound queue and output buffers will also be reset.
INIT	Performs all initialization required to be done once.
INSERTQ	Inserts a three byte MIDI packet into queue.
MAINROT	Solves main rotor equations used in calculating the main rotor forces and moments generated due to a given collective and cyclic pitch, and to given helicopter angular and translational rates. This includes the calculation of lift and drag coefficients, coning angle and flap angle rates, and total main rotor forces and moments including ground effect influences.
MIDIEND	Turns off all sounds (all notes off).

Subroutines	Description
MOTION	Performs the motion base driver functions which use motion washout techniques.
ONED	Performs scaling recursion function for DAC values.
PRINTSUB	Prints variables recorded in the real-time file.
QUAT	Computes the quaternions from body rates and accelerations using local linearization.
RECDRS	Plots washout variables on strip chart recorders.
REMOVEQ	Removes a three byte MIDI packet from queue.
S61MB	Performs main S-61 helicoptor subroutine functions.
SAS	Evaluates stability augmentation system for the S-61 helicopter. This subroutine contains the equations used to evaluate the stability argumentation system transfer functions and to calculate the corrections made to the control inputs.
SAS2	Evaluates stability augmentation system for an AH-1S helicopter (Army Cobra S-model).
SAS2IC	Initializes AH-1S helicopter SAS values.
SETUPXT	Initializes the XTRIM array to the starting values of the trim during the trim iteration.
SETUPFT	Equates FTRIM array to the trim equation during the trim iteration.
SIMEQNS	Solves a system of simultaneous linear equations by the Gauss-Jordan reduction method within the trim circuit algorithm.
SOUND	Calculates data required for sound support.
SOUNDHWIC	Manually re-initializes the sound hardware interface.
TAIL	Solves tail equations used to calculate the forces and moments generated due to the tail rotor and the horizontal and vertical stabilizers. General tail velocities are also computed.

Subroutines	Description
TRIMDER	Executes all subroutines necessary to compute the trim equations.
TRMLITE	Compares a control to a tolerance value and sets a console white litght if within tolerance, and sets appropriate increase or decrease cockpit discretes if out of tolerance.
TURRET	Initializes and drives the turret displays.
WINDS	Calculates steady wind components along body axes which, if selected, may be added to the body velocities.
XMATRIX	Used by trim circuit algorithm, initializes a matrix of independent trim parameters with the first column being the XTRIM, the second being XTRIM with a pertubation on the first element, etc.
XVAL	Used by trim circuit algorithm, initializes the XTRIM array used to generate (N+1) equations where N is the number of trim equations.

PROGRAM USAGE

This section describes the operation procedures, the program setup, including the procedures for obtaining a static check case and dynamic check case, a cockpit checkout, and console operation procedure for the S-61 real-time simulation program.

OPERATION PROCEDURES

The S-61 real-time application program libraries and object files are loaded and linked to generate a real-time absolute (executable) file (s61exec). A generated Symbolic Memory Map (SMM) table, which contains all of the S-61 program variables and their addresses, is transferred from the CONVEX computer to the Real-Time System Console (RTSC) PC[7] within the Real-Time Control Console (RTCC) (see figure 4). Execution of the 'rts' program on the RTSC then links the control console touch screen with the real-time absolute file for dynamic program interaction. The 'rts' program requires two arguments: the first identifies the configuration file that specifies the control console screen format (s61.dat) and the second identifies the SMM (smm.out).

Scheduling is then performed to allocate all necessary resources (i.e., console number, highway, CGI eye point, VMS site, CRDS number, etc.). Finally, the real-time program is executed and the "RESET" mode switch is activated on the RTSC touch screen (see figure 5). Highway de-scheduling is taken care of by the Supervisor at logout.

Following the execution of the RTSC program and the scheduling of required resources, the S-61 real-time application program is executed on the CONVEX machine. An illustration of these steps are included in Appendix A.

[7] 80486 (UNIX) PC running SCO (Santa Cruz Operation) Open Desk Top.

PROGRAM SETUP

Once real-time status is achieved, the real-time analyst may proceed with the program setup using the S-61 console touch screen format on the RTSC (see figure 6).

Step 1: **"Send CGI Data" ON** -- if a CGI eye point was scheduled
Step 2: **"Stop CRD Data" ON** -- if no CRDS was scheduled

Any time after the program setup, the analyst may conduct either a static or dynamic check case. In performing a static check case, proceed with the following steps:

Step 1: **"S61 SAS Model" OFF**
Step 2: Establish static check case conditions by entering the following command on the RTSC status line: **c KNOTS 6**
Step 3: Activate **OPERATE, RESET,** and **PRINT** switches respectively.

The PRINT switch will print variables that were written to the real-time disk during OPERATE mode to a file named 'CHECKCASE' and to a remote printer. Table I.A. illustrates values to be expected during a daily static check case run.[8]

In performing a dynamic check case, proceed with the following steps:

Step 1: Establish dynamic check case conditions by entering the following commands on the RTSC status line

1) **c MCASE 3** -- Sets initial conditions with an airspeed of 80 knots and an initial altitude of 5000 feet
2) **c NT 1** -- number of iterations in which to write to the real-time disk (32 times/sec @32 Hertz; 31.25 ms frame time)
3) **c IDYNCK 200** -- Series of doublet input on control blade angles
   or
   c IDYNCK 100 -- Series of doublet input on control sticks[9]
4) **v t** -- View time

Step 2: Optional: to use strip charts

1) **c CHARTON T**
2) Set recorder speed (5 mm/s, 10 mm/s, ..., 250 mm/s, etc.)
3) Turn strip charts on

Step 3: **"Trim" ON** -- observe that the white light 1 [WLITE(1)] illuminates if a 'Trim good' was achieved and FSS(5) will turn back OFF.
Step 4: **"Dynamic Check" ON**

---

[8] Table I.B. illustrates values to be expected during a full validation static check case run.

[9] IDYNCK option 100 is mostly used as an input test to be compared with flight test data.

Step 5: Activate **OPERATE** for 20 seconds then press **RESET** and **PRINT** switches, respectively.

The PRINT switch will print variables that were written to the real-time disk during OPERATE mode to a file named 'CHECKCASE' and to a remote printer. Table II.A. and Table II.B. illustrate a graphical representation of the values to be expected (using IDYNCK options 200 and 100, respectively) during a dynamic check case run. These graphs were generated with the use of another data file named 'OUTPUT' and a *sifplot* utility. The *sifplot*[10] utility has been tailored for plotting data generated during real-time simulation sessions.

Before proceeding with routine real-time operations, proceed with the following steps to reinitialize the program setup:

Step 1: Enter the following commands on the RTSC status line:
1) **c MCASE 1** -- Sets up initial conditions
2) **c NT 32** -- Write to the real-time disk every 32 iterations (1 time/sec @32 Hertz; 31.25 ms frame time)
3) **"Dynamic Check" OFF**

Step 2: Optional: if strip charts were used
1) **c CHARTON F**
2) Turn strip charts off

With the successful completion of the static and dynamic check cases, the analyst is ready to conduct the cockpit checkout. If no cockpit was scheduled, the analyst may conduct test cases from the RTCC (see figure 4).

[10] Details on *sifplot* utility may be found in reference 4.

COCKPIT CHECKOUT

Communication between the VMS site and the RTCC is established through an intercom system. After a successful static and dynamic check cases are performed, the simulation is ready for the cockpit checkout. The cockpit checkout consists of performing a static cockpit instrument checkout. This static checkout allows the VMS site technician and the analyst to verify that all voltage signals received by the VMS are valid before further operations commence.

The Cockpit Checkout procedure is as follows:

Step 1: **"Static" ON** -- deactivate once VMS site technician statically checks and verifies correct data is being received by the VMS site hardware. Table III.A. illustrates values to be expected during the static cockpit instrument checkout.
Step 2: **"Cockpit Control" ON** -- to receive inputs from the cockpit
Step 3: Site technician performs control deflection inputs and verifies values with real-time analyst. Table III.B. illustrates values to be expected during the cockpit control deflection checkout.[11]

With a successful cockpit checkout, the simulation is ready for piloted test flights.

CONSOLE OPERATION PROCEDURE

Upon completion of the program setup and cockpit checkout, the simulation program is ready to have initial condition values established for a piloted test flight (i.e., heading, altitude, or forward velocity).

The console operation procedure is as follows:

Step 1: Optional: Select desired initial conditions (default: MCASE = 1)
MCASE : 1 -- 12 ft / Hover / align with runway 26L WEST
2 -- 0 ft / Hover
3 -- 800 ft / 80 Knots
4 -- 272 ft / 80 Knots
5 -- 350 ft / 80 Knots
6 -- 350 ft / 80 Knots / runway approach
Step 2: **"Cockpit Control" ON** -- activates subroutine ADCIN which allows control stick inputs to the real-time program
Step 3: **"S61 SAS Model" ON** -- activates the S-61 SAS subroutine
Step 4: **"Trim" ON**

---

[11] Table III.C. illustrates control deflection voltage scale used by the simulation application to represent these values.

Step 5: Pilot will proceed to set potentiometers 1, 2, 3, and 4 on the cockpit instrument panel until white lights 21 through 24 [WLITE(21...24)] illuminate on the RTSC touch screen, respectively; indicating that the controls are trimmed

Step 6: Wait for VMS site technician to notify analyst that motion base control has been transferred to the RTSC

Step 7: Select "**HOLD**" mode switch for approximately 15 seconds

Step 8: Select "**OPERATE**" mode switch

Caution: When using motion ALWAYS remain in HOLD for at least 15 seconds in order to allow the motion base to trim to the flight conditions.

To reinitialize the sound driver, proceed by entering the following commands on the RTSC status line:

Step 1: **c NOSOUND T**

Step 2: **c LSRESET T**

Step 3: **c NOSOUND F**

Note that step 3 may be skipped if no sound is required.

To terminate execution of the application program, proceed with the following steps:

Step 1: Confirm that VMS technician is in "local" control mode

Step 2: Select "**TERM**" mode switch on the RTSC touch screen

Step 3: Logout from all scheduled machines (i.e.: CRDS, RTSC, etc.)

PROGRAM VALIDATION

Since the software conversion of the S-61 real-time application program from the CDC CYBER 175 computer to the current FSCS CONVEX C3800 supercomputer, continuous validation checks have been conducted by means of static and dynamic check cases to assure that a valid representation of the program application is maintained. The results obtained from these check cases from the CONVEX S-61 real-time program version were compare to previously generated results from the CDC CYBER 175 program version. The closely related results obtained validates the converted version. Any data discrepancy may be attributable to CONVEX machine precision.

In addition, an actual helicopter pilot conducted an evaluation of the S-61 real-time application program in the VMS. The purpose was to evaluate the S-61 simulation flying qualities and the quality of the helicopter sound generated by a state-of-the-art sound simulator. The pilot found the handling qualities of the S-61 simulation adequate for a transport/passenger type helicopter simulation. In addition, the engine/rotor noise cues were found to vary in response to power changes fairly accurately. The sound quality/fidelity generated using the new sound simulator was a great improvement over the previously used analog-based sound generator.

CONCLUDING REMARKS

The S-61 real-time application is a man-in-the-loop real-time helicopter flight simulation program, simulating the Sikorsky S-61, a commercial passenger-type helicopter. With minor changes to the mathematical model and to the detailed aircraft data, this program may be used to simulate a variety of single-rotor helicopters. This application program is integrated with the VMS, a CGI system, and a CRDS provided by the LaRC ARTSS.

The VMS is a six Degrees-Of-Freedom (DOF) synergistic motion base and a visual out-the-window scene with a Field-Of-View (FOV) of 106 by 36 degrees. The VMS cockpit is equipped with a friction-type collective control and programmable control loading systems to provide for pitch, roll, and yaw controls.

With the acquisition of the CONVEX supercomputers, all flight simulation application programs required conversion to the new FSCS. Many improvements have been implemented with the S-61 real-time application program since its conversion. These include: usage of upgraded real-time libraries; removal of dead, obsolete, and/or redundant code; interface to the Denver Stapleton Airport data base on the CGI system (Figure 2); the addition of a new EADI display on the CRDS (Figure 3); interface to a state-of-the-art sound simulator; modifications to the FORTRAN syntax within the source code in order to be compatible with the latest CONVEX FORTRAN 77 compiler; implementation of a quaternion coordinate system; an improved SAS algorithm; and a restructured main program that executes more efficiently during real-time mode.

Prior to implementing these improvements, this conversion process required various essential steps to assure the proper transition to the new computer architecture. Major issues to be concerned with when conducting a similar conversion between systems include:

1. Word Size. As a first step in converting software to another computer, programmers must note the target computer word size and the difference between the word size previously used. The effect of word size must be determined and corrections made accordingly. This may involve compiling the program with a different default data representation.

2. Discrete Bit Format. New computer architecture may bring slight differences in bit format that can be devastating. The programmer must identify all code sections that perform bit manipulation to adjust for differences. Careful attention to communication with the input/output system must be done to ensure that all discretes are packed/unpacked into the proper bit positions.

3. Memory Space. The programmer must determine how the system loads the programs into its allocated memory and must organize called COMMON blocks, subroutines and/or functions accordingly to effectively utilize memory space.

4. Real-time Synchronization. The programmer must be aware of the time constraint involved during the execution of a real-time program and determine the necessary time frame needed to perform tasks effectively else a lost-time synchronization error may occur.

5. Compiler Differences. These may include: file usage (compiler may dictate specific methods which file parameters may be used), syntax, variable declarations and constants, and supported options such as debug and cross-referencing options.

6. Hardware Interface. A new computer architecture may have a new real-time input/output interface that will effect the program interface.

7. Real-time System Supervisor Environment. The programmer must be aware of the real-time system environment to quickly identify problems encountered that conflict with the environment.

8. Loader Controls. The programmer must note all necessary options and default settings needed such as *pre-paged* and *non-swap* options.

9. Linking. The programmer must identify which libraries, hardware and other resources are necessary for either running in Batch or Real-time mode.

10. Program Control Restructure. The conversion may be to a more sophisticated and advanced system which provides more resources. The old program control structure should be

examined to remove obsolete implementations and to use new features offered by the new architecture.

11. Validation. The programmer must continuously conduct validation checks (as in this case by means of static and dynamic check cases) to assure that a valid representation of the program application is maintained.

12. New/Improved Utilities. The new system may provide more powerful utilities that may facilitate either the conversion process or daily operation of the program. The programmer must be aware of the system utilities available to exploit their use.

13. Error Messages. The programmer must always note when and under what circumstances errors occur and the actions taken to resolve them. Persistence of error occurrence may lead to identifying other problem areas.

14. Program Efficiency. As conversion proceeds, the programmer should note possible enhancements or more effective methods to execute the program such as modularization and removal of dead/obsolete and redundant code. It is wise to implement modifications to program execution after the conversion is completed to avoid compounding problems and delaying conversion deadlines.

Being aware of these 14 major issues during a similar conversion process will allow the programmer to foresee and/or overcome many challenges and increases the likelihood of a successful transition to a new system.

The S-61 real-time application program has not been use in research for several years. However, with its conversion and enhancement, interest has been demonstrated for its use in conducting helicopter drop model test flight training.[12]

Langley Research Center, National Aeronautics and Space Administration,
Hampton, Virginia, July 16, 1993.

---

[12] Refer to reference 5 to obtain information regarding procedures for initiating and conducting research through real-time flight simulation with this or any other available flight simulation application program.

APPENDIX A - Operational Procedures

The following are options provided by a UNIX script file named "s61load" (followed by parts of the script source code) and steps that illustrate how various real-time operational procedures are accomplished for the S-61 real-time application program.

1. Loading and linking program application on CONVEX in real-time mode:

a) **~mart/S61MB/s61load load**<ENTER>

```
set SCRATCH = /scr/mart
echo "$SIMID Real-time library loading in progress ... "
ld -X -fn -o $SCRATCH/s61exec /usr/lib/crt/crt0.o \
   -u _MAIN__ \
   -Eprepage -Enonswap \
   -A_use_libc_sema=___ap\$use_libc_sema \
   -A__iob=___ap\$iob \
   -lcgi -lrts $SRCLIB $GWLIB $WASHHLIB  -lsuper -lrt -lvm \
   -m -lU77 -lF77 -lI77 -lD77 -llfs -lm -lg -lmathC2 -lc \
   -Mhlos >!  $SCRATCH/s61.ldmap
ls -al  $SCRATCH/s61*
```

2. Generation of SMM and transfer to RTSC (console 1 => rtsc1):

a) **~mart/S61MB/s61load map**<ENTER>

```
echo "Symbolic Memory Map Generation in progress ... "
cat ~mart/S61MB/XREF/*.xrf > SYMMAP.IN
~ral/Scripts/numprogs SYMMAP.IN
/sys/realtime/smm/newsymmap -b  $SCRATCH/s61exec -i SYMMAP.IN -o smm.out
rcp -p smm.out rtsc1:smm.out; ls -al smm.out
echo "Symbolic Memory Map Generation completed."
```

3. Scheduling of necessary resources:

a) **~mart/S61MB/s61load sked**<ENTER>

```
echo "$SIMID real-time scheduling in progress ... "
/sys/realtime/sked -h $hwy -f $RCT -c $siterec -s $SIMID
```

4. RTSC startup:

a) Login on to RTSC1: login:<USER><ENTER>
password:<XXXXXX><ENTER>

b) Execute RTSC program: **rts s61.dat smm.out**<ENTER>

5. Real-time program execution on CONVEX machine:

a) **$SCRATCH/s61exec**<ENTER>

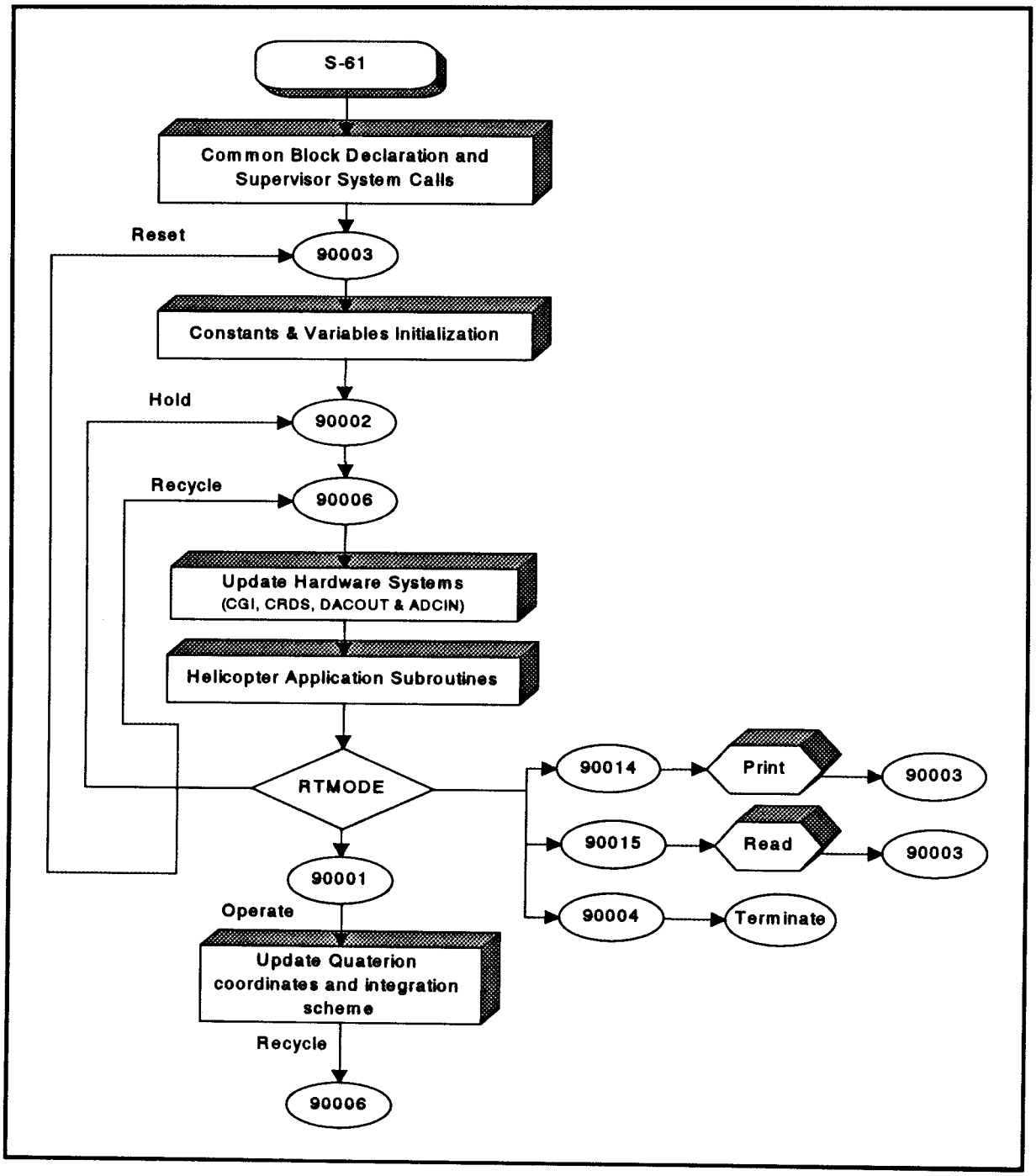
APPENDIX B - Makebatch Script Listing

The following is the source code for a UNIX script file named "makebatch" that illustrates how to load and link the S-61 real-time application program on the CONVEX in batch mode.

```
# #!/bin/csh -f
# DATE       : APRIL 8 1993
# FILE(S)    : makebatch
# PURPOSE    : Script - Used to generate an executable for running in batch.
#              By setting nohwy=1 this will invoke the call to the BATCH routine.
# USAGE      : make -f makebatch
#---------------------------------------------
set SCRATCH = /scr/mart

SOURCES = s61mb.f adcin.f aerovar.f batch.f bodyder.f bulkdat.f \
          cgiintf.f config.f crds61.f dacout.f dynck.f \
          eadi.f earthm.f engine.f eval.f \
          fandm.f fo.f fopq.f fus.f gust.f helitrim.f \
          insound.f init.f  insertq.f \
          mainrot.f midiend.f motion.f oned.f printsub.f quat.f \
          readpixel.f recdrs.f removeq.f\
          sas.f sas2.f sas2ic.f setupft.f setupxt.f simeqns.f \
          tail.f trimder.f trmlite.f turret.f\
          winds.f xmatrix.f xval.f  \

 OBJECTS = $(SOURCES:.f=.o)

s61batch : $(OBJECTS)
        fc -p8 -db -o  $SCRATCH/s61batch $(OBJECTS) -lcgi -lcrds \
        -l /cx/et/msy/roc/geh/Srclibs/srclib5 \
        -lnstub -lrt -lrts -l ~swc/Washlib/washhlib \
        -m -lm -lmathC2 -lc

        @echo "==> Enter ' $SCRATCH/s61batch' to begin batch execution"

.f.o:
        fc -p8 -db -c $<
```

APPENDIX C - High Level S-61 Program Flow Chart

The following contains a high level flow chart which deplicts the overall control structure for the S-61 real-time application program. Note that the numbers contained within the flow chart pertain to user defined FORTRAN label statements (i.e., 90002, 90003, etc.).



APPENDIX D - COMMON Block FORTRAN Listing

The following contains the FORTRAN listing of the S-61 real-time application program COMMON blocks.

```
*comdeck accel.com
      COMMON / ACCEL  / AX    , AY    , AZ
C*
*comdeck advc.com
      COMMON / ADV C  / OMEG, WIM, VT, QV, QL, ASPD
C*
*comdeck advp.com
      COMMON / ADV P  / ADVP1, ADVP2, WIMP
C*
*comdeck cgicomm.com
      COMMON / CGICOMM / IGNUM     , CHNGRPN   , CHNNUM ( 2),
     *    CHNHFOV( 2), CHNVFOV( 2), CHNPSI ( 2), CHNTHE ( 2),
     *    CHNPHI ( 2), CGIINT    , LCHKSUM   , LDEBUG    ,
     *    IRNWAY    , CHANINT    , EYEPNT    , LUMINT    ,
     *    VISINT    , SUNINIT    , USEDCS    , USEPDCS   ,
     *    PDCSON    , DCSNUM ( 7), SPCREQ ( 7), SEQID( 7),
     *    MODSEL ( 7), SUNINT    , FOGINT    , POLYINT   ,
     *    RDUCINT   , LITEMSK    , POLYMSK   , VISDIST   ,
     *    SUNPSI    , SUNTHE     , FOG       , FOGTYPE   ,
     *    FOGVIS    , FOGBOT     , FOGTOP    , CLOUDS    ,
     *    CLDTYPE   , CLDVIS     , CLDBOT    , CLDTOP    ,
     *    LGTNGON   , LGHTING    , LOBE      , LLSEL     ,
     *    LLINT     , SLGON      , SLGNUM    , SLGSEL    ,
     *    SLGINT    , MLGON      , MLGNUM    , MLGARRY(32),
     *    MSEQON    , MSEQDAT( 8), MSEQOFF   , TEXMOTN   ,
     *    MVECNUM   , XTEXVEL    , YTEXVEL   , XTEXOFF   ,
     *    YTEXOFF   , COLDON     , COLDECT   , CDSEQNO   ,
     *    HATON     , HATSEL     , HATDCSN   , HAT       ,
     *    ZHAT      , ERRWFLG    , CGII(160) , CGIO(160) ,
     *    STCGII    , STCCGO     , CGIINDX   , CGIONDX
C*
      INTEGER   CHNGRPN, CHNNUM
      INTEGER   PDCSON , DCSNUM , SPCREQ , SEQID
      INTEGER   SUNINT , FOGINT , POLYINT, RDUCINT, POLYMSK
      INTEGER   FOGTYPE, CLDTYPE, SLGNUM , SLGSEL , SLGINT
      INTEGER   COLDECT, CDSEQNO, HATSEL , HATDCSN
      INTEGER   CGII,CGIO
      INTEGER*4 STCGII,  STCCGO,  CGIINDX, CGIONDX
C*
      LOGICAL   CGIINT , LCHKSUM, LDEBUG , CHANINT, EYEPNT
      LOGICAL   LUMINT , VISINT , SUNINIT, USEDCS , USEPDCS
      LOGICAL   FOG    , CLOUDS , LGTNGON, LOBE   , SLGON
      LOGICAL   MLGON  , MSEQON , MSEQOFF, TEXMOTN, COLDON
      LOGICAL   HATON  , HAT    , ERRWFLG
C*
*comdeck cgizcom.com
      COMMON / CGIZCOM / CGICMD    , CGIPTR    , CGIERR    , CGIMAX    ,
     *           CHNMAX    , CHNMSK    , GRPMAX    , CHKSUM    ,
     *           CHGINT    , CHGLIT    , CHGCOL    , CHGLOB    ,
     *           CHGVIS    , CHGSIZ    , CHGCON    , CGITXF    ,
     *           CDEBUG    , IXYLIM    , IZZMAX    , IZZMIN    ,
     *           L08BIT    , L16BIT    , L32BIT    , CGINIT    ,
     *           ASCALE    , CGIIGN (3), CHNGRP (8), LITGRP (8),
     *           CGIBUF(301)
C*
      INTEGER   CGIIGN , CGICMD , CGIPTR , CGIERR , CGIMAX , CHNMAX
```

```
      INTEGER  CHNMSK , GRPMAX , CHNGRP , CHGINT , CHGLIT , CHGCOL
      INTEGER  CHGLOB , CHGVIS , CHGSIZ , CHGCON , ASCALE
      INTEGER  LITGRP , CGIBUF
C*
      LOGICAL  CHKSUM , CGITXF , CDEBUG , CGINIT
C*
*comdeck cmidi.com
      PARAMETER ( SQMAX=96, MIDIMAX=48, MIDINUM=MIDIMAX/8 )
      PARAMETER ( NOTEON='ON ', NOTEOFF='OFF', NOTEMOD='MOD' )
      PARAMETER ( NUMSNDS=45 )
C
      COMMON / CMIDI / MIDIOUT(MIDINUM), MIDDBUG, MIDIERR, SQHEAD
     *,          SQTAIL , MIDIBC , MIDSTAT, ORDMIDI, SOUNDQ
     *,          BLDSND , ENGSND , CALLDON(NUMSNDS), TESTVOL
     *,          TONES  , NOSOUND, MIDCHAN, ALLOFF , LSRESET
C
      INTEGER*4 ORDMIDI
      INTEGER*1 SOUNDQ(SQMAX), MIDIBYT(MIDIMAX), TESTVOL, MIDCHAN
      INTEGER*1 ALLOFF, BLDSND, ENGSND
      INTEGER  SQHEAD, SQTAIL
      LOGICAL  MIDDBUG, MIDIERR, CALLDON, TONES(13), NOSOUND, LSRESET
C
      EQUIVALENCE (MIDIOUT(1), MIDIBYT(1))
C*
*comdeck cntlc.com
      COMMON / CNTL C  / AOSC  , A1SC  , B1SC  , THTRC
C*
*comdeck cntlp.com
      COMMON / CNTL P  / AOSC0 , A1SC0 , B1SC0 , THTRC0 , XAOSG ,  YCSG  , XCSG
C*
*comdeck cntsas.com
      COMMON / CNTSAS  / AOSCS , A1SCS , B1SCS , THTRCS
C*
*comdeck coef.com
      COMMON / COEF    / CT, CP, CQ, MU, MTIP, VS
C*
      REAL     MU    , MTIP
C*
*comdeck console.com
*
**    SYSTEM COMMON BLOCK - USED TO COMMUNICATE W/CONSOLE & ECC
*
      COMMON / CONSOLE / CADC(12) , CDAC(36), CIDIS( 2),  CODIS( 2), VBATCH,  RAM0(200),
     *           RAM2(200), RAMN
C*
      REAL*8   CADC, CDAC
      INTEGER*8 CIDIS, CODIS,VBATCH
      INTEGER*2 RAM0, RAM2
      INTEGER*1 RAMN
*
**    CADC  - CONSOLE ADC'S (6 SLIDE POTS, 4 ROTARY POTS, 2 X-Y JOYSTICK)
**    CDAC  - CONSOLE DAC'S (FOUR 8 CHANNEL STRIP CHARTS, 2 X-Y PLOTTERS)
**    CIDIS - CONSOLE INPUT DISCRETES (VARIOUS ECC BUTTONS)
**    CODIS - CONSOLE OUTPUT DISCRETES (STRIP CHART MOTOR DRIVES/EVENT MARKERS)
**    RAM0  - RAM INPUT (CONVEX TO RTSC/CAMAC)
**    RAM2  - RAM OUTPUT (RTSC/CAMAC TO CONVEX)
**    VBATCH - ZERO IF BATCH, ELSE NON-ZERO
**             - SET BY REAL-TIME SUPERVISOR
*
C*
     COMMON / CFSS   / NFSS  , FSS(24)
*
**    SYSTEM COMMON BLOCK - USED TO COMMUNICATE W/THE FSS'S ON THE RTSC 12
```

```
*
**     RTSC FSS SWITCHES:
**     ----------------------
**     FSS(1)  ==> DYNAMIC
**     FSS(2)  ==> GUSTS
**     FSS(3)  ==> DISPOSE TO MAIN PRINTER
**     FSS(4)  ==> WINDS
**     FSS(5)  ==> TRIM
**     FSS(6)  ==> n/a
**     FSS(7)  ==> BIAS CONTROLS
**     FSS(8)  ==> STATICS
**     FSS(9)  ==> S-61 SAS MODEL
**     FSS(10) ==> COBRA SAS MODEL
**     FSS(11) ==> n/a
**     FSS(12) ==> FLIGHT DIRECTOR (not available)
**     FSS(13) ==> COCKPIT CONTROL
**     FSS(14) ==> TURRET
**     FSS(15) ==> STOP CRD DATA
**     FSS(16) ==> SEND CGI DATA
**     FSS(17) ... FSS(24) ==> CGI SETTING CONDITIONS (not shown)
**     FSS(25) ==> SOUND DUMP BUFFER (not shown)
*
C*
      INTEGER*1  NFSS
      LOGICAL*1  FSS
C*
      COMMON / CLITE  / NLITE  , LITE (48)
C*
*
**    SYSTEM COMMON BLOCK - USED TO COMMUNICATE W/THE LITE'S ON THE RTSC 12
*
      INTEGER*1  NLITE
      LOGICAL*1  LITE
C*
      LOGICAL*1  WLITE(39)
      LOGICAL*1  RLITE( 8)
      EQUIVALENCE ( LITE ( 1), WLITE(1) )
      EQUIVALENCE ( LITE (41), RLITE(1) )
C*
      COMMON / CMCP   / NMCP   , MCP(10)
C*
*
**    SYSTEM COMMON BLOCK - USED TO COMMUNICATE W/THE MCP'S ON THE RTSC 12
*
**      RTSC MCP SWITCHES:
**      ----------------------
**     MCP(1)  ==> OPERATE
**     MCP(2)  ==> HOLD
**     MCP(3)  ==> RESET
**     MCP(4)  ==> IDLE
**     MCP(5)  ==> TERM
**     MCP(6)  ==> ERASE1
**     MCP(7)  ==> ERASE2
**     MCP(8)  ==> PANEL REFRESH
**     MCP(9)  ==> PRINT
**     MCP(10) ==> READ
C*
      INTEGER*1  NMCP
      LOGICAL*1  MCP
C*
*comdeck contrl.com
      COMMON / CONTRL  / AOS   , A1S   , B1S   , THTR  , OMEGD
C*
```

```
*comdeck crdcomm.com
    COMMON / CRDCOMM / CRDBUF(500), ORDCRDO, ORDCRDI,INRTB,  EADCHAN,  EADSCOP,  EADBLOK, CRDO(500)
C*
    INTEGER*4  CRDBUF    , ORDCRDO, ORDCRDI
    INTEGER    EADCHAN   , EADSCOP, EADBLOK, CRDO, INRTB
C*
*comdeck crecdr.com
    COMMON / C RECDR / RECMIN(32), RECMAX(32)
C*
*comdeck ctrim.com
    COMMON / C TRIM  / XTRIM(11), FTRIM(11), QVEC (12)
C*
*comdeck deflect.com
    COMMON / DEFLECT / XAOSCOL, XCSCOL , YCSCOL , XTRCOL,  XCST  , YCST  , XTRT  , XAOST ,
   *                           XCOL  ,  XLAT  , XLON  , XRUD  , PEDPOS
C*
*comdeck dircos.com
    COMMON / DIRCOS  / C11   , C12   , C13   , C21   , C22   ,  C23   , C31   , C32   , C33
C*
*comdeck display.com
*
**   SYSTEM COMMON BLOCK - TURRET LABELS
*
    COMMON / DISPLAY / NEXTPIC     , LSTPSNT  , PDP     ,
   *            PLOGL0(4)    , PLOGL1(2) , PLOGL2(6) , PLOGL3(6) ,
   *                    PLOGV1(2) , PLOGV2(6) , PLOGV3(6) ,
   *            PDPXCORD     , PDPYCORD ,
   *            STATINTT     , STO03    ,
   *            STO06        , STO11    , STO20    ,
   *            STO39        , STO83    , STO259   ,
   *            CMDNMBR      , BYTRCMD  , IDXORDR  ,
   *            PDPFOR( 4)   , PDPCMD(11), PDPPICT(20,80),
   *            NUMLINS(35)  ,
   *            NOCMND       , FLSHPDP  , STPFLSH  ,
   *            CLERPDP      , DIMPDP   ,
   *            TRTO39(39)   , TRTO20(20), TRTO11(11),
   *            TRTO08( 8)   , TRTO03( 3), TRTO83(83),
   *            TRTO259(259) , TRRTI01  ,
   *            READPIC      , TURRETON ,
   *            PDP6CHR      , PDP3CHR
C
C*   ESTABLISH INDICES FOR PICTURE DATA
C
    INTEGER*6 NEXTPIC, LSTPSNT, PDP
C
C*   ESTABLISH CHARACTER DATA
C
    CHARACTER*6 PDP6CHR
    CHARACTER*3 PDP3CHR
C
C*  ESTABLISH DATA ARRAYS FOR TURRET LABEL DISPLAYS
C
C*  PLOGL0 - ARRAY HOLDING HOLLERITH OR INTEGER VALUES FOR
C*         CONSOLE LABEL DISPLAY.
C*  PLOGL1 - ARRAY HOLDING HOLLERITH OR INTEGER VALUES FOR
C*         CONSOLE PROGRAMMABLE PUSH BUTTON DISPLAYS
C*  PLOGL2 - ARRAY HOLDING HOLLERITH OR INTEGER VALUES FOR
C*         CONSOLE SLIDE POT LABLES
C*  PLOGL3 - ARRAY HOLDING HOLLERITH OR INTEGER VALUES FOR
C*         CONSOLE ROTARY POT LABLES
C*  PLOGV2 - ARRAY HOLDING HOLLERITH OR INTEGER VALUES FOR
C*         CONSOLE SLIDE POT VALUES
C*  PLOGV3 - ARRAY HOLDING HOLLERITH OR INTEGER VALUES FOR
```

```
C*        CONSOLE ROTARY POT VALUES
C
      INTEGER*8 PLOGL0 , PLOGL1 , PLOGL2 , PLOGL3
      INTEGER*8 PLOGV1 , PLOGV2 , PLOGV3
C
C*   ESTABLISH INPUT DATA READY INDICATOR
C
      INTEGER*4 STATINTT, STO03, STO08, STO11, STO20, STO39, STO83, STO259
C
C*   ESTABLISH X&Y COORDINATE INDEXES
C
      INTEGER*2 PDPXCORD, PDPYCORD
C
C*   ESTABLISH HEADER MESSAGE DATA WORDS
C
      INTEGER*1 CMDNMBR, BYTRCMD, IDXORDR
      INTEGER*1 PDPFOR , PDPCMD, PDPPICT, NUMLINS
      INTEGER*1 NOCMND , FLSHPDP , STPFLSH, CLERPDP, DIMPDP
C
C*   ESTABLISH THE BUFFERS THAT ARE ACTUALLY OUTPUT BY THE SUPERVISOR
C
      INTEGER*1 TRTO259 , TRTO83 , TRTO39 , TRTO20 , TRTO11 , TRTO08, TRTO03
C
C*   ESTABLISH THE BUFFERS THAT ARE INPUT FROM THE TURRET
C
      INTEGER*1 TRRTI01
C
C*   ESTABLISH LOGICAL TO INDICATE FROM CONSOLE THAT PICTURE DATA MAY BE READIN
C
      LOGICAL*1 READPIC
C
C*   ESTABLISH TURRET ON/OFF LOGICAL AND CONSOLE UPDATEING FLAGS
C
      LOGICAL*1 TURRETON
C*
*comdeck dscrete.com
      COMMON / DSCRETE / IDIS (4), ODIS (3)
C*
      INTEGER   ODIS
*
**    IDIS  - (Input)  PACKED DISCRETES FROM THE VMS COCKPIT
**    ODIS  - (Output) PACKED DISCRETES TO THE VMS COCKPIT
*
      EQUIVALENCE ( IDIS(1), ID1    )
      EQUIVALENCE ( IDIS(2), ID2    ) ! *** for future expansion ***
      EQUIVALENCE ( IDIS(3), ID3    ) !    "  "    "
      EQUIVALENCE ( IDIS(4), ID4    ) !    "  "    "

      EQUIVALENCE ( ODIS(1), OD1    )
      INTEGER   OD1
      EQUIVALENCE ( ODIS(2), OD2    ) ! *** for future expansion ***
      INTEGER   OD2
      EQUIVALENCE ( ODIS(3), OD3    ) !    "  "    "
      INTEGER   OD3
C*
*comdeck ederint.com
      REAL*8  N, NDOT
      EQUIVALENCE (DERINT(1, 1), P     ), (DERINT(2, 1), PDOT  )
      EQUIVALENCE (DERINT(1, 2), Q     ), (DERINT(2, 2), QDOT  )
      EQUIVALENCE (DERINT(1, 3), R     ), (DERINT(2, 3), RDOT  )
      EQUIVALENCE (DERINT(1, 4), U     ), (DERINT(2, 4), UDOT  )
      EQUIVALENCE (DERINT(1, 5), V     ), (DERINT(2, 5), VDOT  )
      EQUIVALENCE (DERINT(1, 6), W     ), (DERINT(2, 6), WDOT  )
```

```
      EQUIVALENCE (DERINT(1, 7), N    ), (DERINT(2, 7), NDOT  )
      EQUIVALENCE (DERINT(1, 8), E    ), (DERINT(2, 8), EDOT  )
      EQUIVALENCE (DERINT(1, 9), H    ), (DERINT(2, 9), HDOT  )
C*
*comdeck einteg.com
      COMMON / EINTEG / NT, NTPRINT, INTSCME, KNOTS, MCASE, IDYNCK, ITER
C*
*comdeck elogic.com
      COMMON / ELOGIC /  LOGIC(100)
C*
      LOGICAL LOGIC
C*
      EQUIVALENCE ( LOGIC( 3), TAXI       )
      LOGICAL  TAXI
      EQUIVALENCE ( LOGIC( 6), AUTOH      )
      LOGICAL  AUTOH
      EQUIVALENCE ( LOGIC( 7), AUTOHW     )
      LOGICAL  AUTOHW
      EQUIVALENCE ( LOGIC( 8), MBREADY    )
      LOGICAL  MBREADY
      EQUIVALENCE ( LOGIC(12), HOLDFLG    )
      LOGICAL  HOLDFLG
      EQUIVALENCE ( LOGIC(10), VERTVEL    )
      LOGICAL  VERTVEL
      EQUIVALENCE ( LOGIC(98), CHARTON    )
      LOGICAL  CHARTON
      EQUIVALENCE ( LOGIC(99), IPRINT     )
      LOGICAL  IPRINT
      EQUIVALENCE ( LOGIC(100), NERR      )
      LOGICAL  NERR
C*
*comdeck engc.com
      COMMON / ENG C  / QMAX, QE, SHP, GRWT, OMEGDT, QED
C*
*comdeck engp.com
      COMMON / ENG P  / IROT  , CKE1
C*
      REAL    IROT
C*
*comdeck etable.com
      COMMON / ETABLE /  AOS0,  A1S0, B1S0, THTR0,  U0, V0, W0, PSI0,
     * PHI0, THETA0, UKNOTS, HDOTD0, H0, OMEG0, CG, GRWT0, P0, Q0, R0, N0,
     * E0, AOSLIM(2), A1SLIM(2), B1SLIM(2), THTRLIM(2),
     * LAPLAC0, SIGMAU0, PSIWD0, VWIND0, TIM(5), AMP

      REAL N0
      REAL LAPLAC0
C*
*comdeck eulcs.com
      COMMON / EUL CS  / COSFI  , SINFI  , COSTH  , SINTH  , COSSI  , SINSI
C*
*comdeck euler.com
      COMMON / EULER / PHI, THETA, PSI,PHIDOT,THEDOT,PSIDOT, PHIDEG, THEDEG, PSIDEG, ALFFDEG, BETFDEG,
     *                 AB1, AB2, AB3, AB4
C*
*comdeck fcnname.com
      COMMON / FCNNAME / CLF1  , CMF1  , CNF1  , CX1   , CX2   ,  CYTR  , CYF   , CZF   , CXW   , CZW   ,
     *           CYVS  , CZHS  , CDSIY , CLSIY , DW    ,  GEV   , GEH
C*
*comdeck file1.com
      COMMON / FILE1  / BUF1(2049), VTAB(100)
C*
*comdeck fnmc.com
```

```
      COMMON / F N M C / XA, YA, ZA, LA, MA, NA
C*
      REAL    LA    , MA    , NA
C*
*comdeck fnmp.com
      COMMON / F N M P / DX    , DXHS  , DXTR  , DXVS  , DZ    ,  DZTR  , DZVS  , DXW   , DZW   , IR
C*
      REAL    IR
C*
*comdeck focom.com
      COMMON / FOCOM  / HFO   , NFO   , AFO(5,1)
C*
*comdeck ftrmpls.com
      COMMON / FTRMPLS / EQLMR  , EQAOSS , EQWIM  , HDOTD
C*
*comdeck funcs.com
      COMMON / FUNCS  / F001(28), F002(28), F003(14), F004(14),
     *          F005(14), F006(28), F007(14), F008(28),
     *          F009(14), F010(28), F011(35), F013(07),
     *          F014(07), F015(07), D018(27), D001(09),
     *          D002(09), D003(09), D004(09), D005(09),
     *          D006(09), D007(09), D008(09), D009(09),
     *          D010(09), D011(09), D013(09), D014(09),
     *          D015(09), D016(18), D017(18), F016(35,10),
     *          F017(35,10),  F018(7,3,4)
C*
*comdeck fusc.com
      COMMON / FUS C  / XF    , YF    , ZF    , LF    , MF    ,  NF    , XW    , WIWM  , ZW    , ALFF  , BETF  , ALFW
C*
      REAL    LF    , MF    , NF
C*
*comdeck fusp.com
      COMMON / FUS P  / CLF2  , CMF2  , CNF2  , CKRF  , CKWIWM , IW    , VLF   , VMF   , VNF   , SXF1  ,
     *          SXF2  , SYF   , SZF   , SXW   , SZW
C*
      REAL    IW
C*
*comdeck gparam.com
      COMMON / GPARAM  / DELT  , RHO   , G     , PI    , PIINV  ,  REVTRAD, DEG2RAD, RAD2DEG
C*
*comdeck intcomm.com
      COMMON / INTCOMM / T     , DT    , INT   , NEQ   ,  ISCHEME,  DERINT(2,9)
C*
*
**    SYSTEM COMMON BLOCK - USED FOR INTEGRATION BY "IGRATE"
*
**    T       - TIME, THE INDEPENDENT VARIABLE IN SECONDS
**    DT      - INTEGRATION STEP-SIZE
**    INT     - INTEGRATION FLOW CONTROL PARAMETER (MUST BE ZERO TO
**              START
**    NEQ     - NUMBER OF INTEGRATIONS PAIRS
**    ISCHEME - USED BY "IGRATE1" TO SPECIFY WHICH INTEGRATION SCHEME
**                TO USE (CAN JUST CALL IGRATE6)
**    THE REMAINING VARIABLES IN THE COMMON BLOCK "INTCOMM" ARE THE
**    VARIABLES TO BE INTEGRATED AND THEIR DERIVATIVES
*
C*
*comdeck lmode.com
*
**    SYSTEM COMMON BLOCK - USED FOR LATCHING MCP WHEN RUNNING BATCH (TRANSPARENT TO USER IN REAL-TIME)
*
      COMMON / LMODE  / LMCP (10)
C*
```

```
      LOGICAL*1  LMCP
C*
*comdeck mask.com
*
**    SYSTEM COMMON BLOCK - USED FOR CONVIENT MASKING
*
      COMMON / MASK   / TMASK64(64), FMASK64(64), TMASK32(32), FMASK32(32),  TMASK16(16), FMASK16(16),  TMASK8(8),
     *                          FMASK8(8),  MTYPE(21),  MMCP(10)
C*
      INTEGER*8 TMASK64 , FMASK64
      INTEGER*4 TMASK32 , FMASK32
      INTEGER*2 TMASK16 , FMASK16 , MTYPE
      INTEGER*1 TMASK8  , FMASK8
      LOGICAL*1 MMCP
C*
**    TMASK - ARRAY USED TO SET A PARTICULAR BIT (E.G. TMASK(4)
**              WILL HAVE ALL ZEROES EXCEPT FOR BIT 4 (1 IS RIGHTMOST)
**    FMASK - ARRAY USED TO CLEAR A PARTICULAR BIT (E.G. FMASK(4)
**              WILL HAVE ALL ONES EXCEPT FOR BIT 4
**    MTYPE AND MMCP - JUST USED BY THE SUPERVISOR
*
C*
*comdeck mrotc.com
      COMMON / MROT C  / FXR   , FYR   , LMR   , LRH   , MRH   ,
     *           QMR   , AOSS  , ALFSIY , UTSIY , BETSI ,
     *           BDTSI , A1SS  , AD1SS , B1SS  , BD1SS
C*
      REAL    LMR   , LRH   , MRH
C*
*comdeck realtim.com
*
**    USER COMMON BLOCK - USED TO LOOK AT COCKPIT
*
      COMMON / REALTIM / ADC(15), DAC(60), SYNCRO(12),  OPERATE,  HOLD,  RESET
C*
      LOGICAL   OPERATE, HOLD, RESET
C*
*
**    ADC  -  ARRAY OF ANALOG TO DIGITAL SIGNALS FROM THE COCKPIT
**    DAC  -  ARRAY OF DIGITAL TO ANALOG SIGNALS TO THE COCKPIT
*
C*
*comdeck rcoeff.com
      COMMON / RCOEFF  / CKL(20), CKD(20), CKQ(20), CKDL(20), CKQL(20),
     *           CKM(20), Y(20)  , WF(20)
C*
*comdeck rotcon.com
      COMMON / ROTCON  / YPE(20), YTW(20), WIF(20), BOVN
C*
*comdeck rotp.com
      COMMON / ROT P   / TWIST , FMRS  , CKAO  , CKAB1 , CKAB2 ,  CKAB3 , NRAD  , NAZ   , NB    , RB    ,   EFH
C*
*comdeck rtbufs.com
*
**    SYSTEM COMMON BLOCK - USED BY SUPERVISOR AS A SCRATCH BUFFER
*
      PARAMETER(NRTBUFC=2000)
      COMMON / RTBUFS  / RTBUFC(NRTBUFC)
      REAL*4 RTBUFC
C*
*comdeck runway.com
      COMMON / RUNWAY /  PSIR, PSIRWY, SXRWY, SYRWY
C*
```

```
*comdeck sas2com.com
      COMMON / SAS2COM / PA  (3), QA1 (3), QA2 (3), A1  (3), RA  (3),
     *            RAP (3), P1B (3), P2B (3), Q1B1(3), Q1B2(3),
     *            Q2B1(3), Q2B2(3), B1  (3), B2  (3), B3  (3),
     *            R1B (3), R1BP(3), R2B (3), R2BP(3), P1C (3),
     *            Q1C1(3), Q1C2(3), C1  (3), C2  (3), C3  (3),
     *            C4  (3), R1C (3), R1CP(3), R2C (3), R2CP(3),
     *            R3C (3), R3CP(3), R4C (3), R4CP(3), ZLIM(3)
C*
      EQUIVALENCE ( P2C   , P1B   )
      DIMENSION P2C   (3)
      EQUIVALENCE ( Q2C1  , Q1B1  )
      DIMENSION Q2C1 (3)
      EQUIVALENCE ( Q2C2  , Q1B2  )
      DIMENSION Q2C2 (3)
      EQUIVALENCE ( P3C   , P2B   )
      DIMENSION P3C  (3)
      EQUIVALENCE ( Q3C1  , Q2B1  )
      DIMENSION Q3C1 (3)
      EQUIVALENCE ( Q3C2  , Q2B2  )
      DIMENSION Q3C2 (3)
      EQUIVALENCE ( P4C   , PA    )
      DIMENSION P4C  (3)
      EQUIVALENCE ( Q4C1  , QA1   )
      DIMENSION Q4C1 (3)
      EQUIVALENCE ( Q4C2  , QA2   )
      DIMENSION Q4C2 (3)
C*
*comdeck sasinpt.com
      COMMON / SASINPT / SASCON1, SASCON2, SASCON3, SASCON4, SASCON5,
     *            SASCON6, SASCON7, SASCON8, LAPLACE, THETAPR,
     *            PHIPR  , A1SCPR , B1SCPR , GAINLAP, HHGAIN
C*
      REAL    LAPLACE
C*
*comdeck scales.com
      COMMON / SCALES  / SF1   , SF2   , SF3   , SF4   , SF7   , LOGHH , PSIHH
C*
      LOGICAL  LOGHH
C*
*comdeck scpsir.com
      COMMON / SCPSIR  / SNSIR(4),  COSSIR(4)
C*
*comdeck shipp.com
      COMMON / SHIP P  / IXX   , IYY   , IZZ   , MASS
C*
      REAL    IXX   , IYY   , IZZ   , MASS
C*
*comdeck skeds.com
      COMMON / SKEDS / CGISKED, CRDSKED, CABON, MIMON
C*
      INTEGER CGISKED, CRDSKED
C*
      LOGICAL CABON, MIMON
C*
*comdeck sswtchs.com
*
**   USER COMMON BLOCK - USED IN HARDWARE CONFIGURATION
**    ISSWTCH( 1 = CONSOLE, 2 = CRD, 3 = CGI, 4 = MIM, 5 = COCKPIT,  6 = n/a)
*
     COMMON / SSWTCHS / ISSWTCH(6)
C*
*comdeck sticks.com
```

```
      COMMON / STICKS  / XAOS   , YCS    , XCS    , XTR
C*
*comdeck summs.com
      COMMON / SUMMS   / SXDR, SYDR, SQDR, SLDR, DPDR, LPDR, MPDR, QPDR, THTRP, VTRP
C*
*comdeck swind.com
      COMMON / SWIND   / PSIWR  , VWIND  , UW     , VW     , WW
C*
*comdeck tailc.com
      COMMON / TAIL C  / ALFHS  , BTVS   , ZHS    , YVS    , VTR    , WHS    , YTR    , DELE
C*
*comdeck tailp.com
      COMMON / TAIL P  / CKTR1, CKTR2, IHS, IVS, SZHS, SYVS
C*
      REAL     IHS    , IVS
C*
*comdeck trimat.com
      COMMON / TRIMAT  / XMAT(11,12), EMAT(12,13), TRSUM(12), NEQN   ,   NEQP1   , NEQP2     , XMATG(11)
C*
*comdeck userinfo.com
*
**    SYSTEM COMMON BLOCK - SUPERVISOR STATUS
*
            COMMON /USERINFO/ USR, RCT, FT, TIMREM, TIMUSED,
     *           BEGCLK, LSTIME, SRTF, MRTF, BKGNDF, CLASS,
     *           ERRCOUNT, FRAMECT
*
**    USR     - NAME OF COMMON BLOCK
**    RCT     - REQUESTED COMPUTE TIME IN MICROSECONDS
**    FT      - FRAME TIME IN MICROSECONDS
**    TIMREM  - CPU TIME NOT USED PREVIOUS FRAME
**    TIMUSED - CPU TIME USED PREVIOUS FRAME
**    BEGCLK    - WALL CLOCK TIME AS OF START OF FRAME
**    LSTIME    - LOST TIME SYNCHRONIZATION FLAG, 1 IF SET FOR CONTINUOUS
**                LOST SYNCHRONIZATION RECOVERY, 0 IF NOT
**    SRTF      - SRT FLAG, 1 IF SRT, 0 IF NOT
**    MRTF      - MRT FLAG, 1 IF MRT, 0 IF NOT
**    BKGNDF    - FLAG INDICATION BACKGROUND PROCESSING IS ENABLED
**    CLASS     - FLAG FOR REAL TIME LIBRARY TYPE
**    ERRCOUNT - FOR COUNTING ERRORS, CURRENTLY USED FOR CHG/DISP
**    FRAMECT   - NUMBER OF FRAMES, INCLUDES FRAMES NOT SEEN BECAUSE OF
**                LOST SYNCHRONIZATION RECOVERY BY SHIP
*
            INTEGER*8  TIMREM, TIMUSED, BEGCLK
            CHARACTER*8 USR
            INTEGER*4 RCT, FT, FRAMECT
            INTEGER*2 LSTIME, SRTF, MRTF, BKGNDF, CLASS, ERRCOUNT
C*
*comdeck wcom.com
      COMMON / WCOM    / TABW(131)
C
C*    EQUIVALENCE MOTION LEGS TO WASHOUT VALUES
C
      EQUIVALENCE (TABW(118), EX1)
      EQUIVALENCE (TABW(119), EX2)
      EQUIVALENCE (TABW(120), EX3)
      EQUIVALENCE (TABW(121), EX4)
      EQUIVALENCE (TABW(122), EX5)
      EQUIVALENCE (TABW(123), EX6)
C*
*comdeck wind.com
      COMMON / WIND    / UGUST , VGUST , WGUST , UNOISN , VNOISN , WNOISN , SIGMAU
C*
```

```
*comdeck climit.com
*
**   INTRINSIC FUNCTION - Closed LIMITER
*
**    X        = INPUT
**    XMIN     = LOWER LIMIT
**    XMAX     = UPPER LIMIT
**    CLIMIT   = LIMITED INPUT, THE ANSWER
*
      CLIMIT(X,XMIN,XMAX) = AMIN1 (XMAX, AMAX1 (X, XMIN))
C*
```

APPENDIX E - Bulkdat FORTRAN Listing

The following contains the Bulkdat FORTRAN subroutine listing which initializes values for various S-61 real-time application program common blocks variables (See Appendix D).

```
      BLOCK DATA BULKDAT
C*
C*****************************************************************
C*
C*    Description: BLOCK DATA SUBPROGRAM ASSIGNS VALUES FOR
C*    PARAMETERS IN LABELLED COMMON BLOCKS
C*
C*****************************************************************
C*
*CALL ADVP.COM
*CALL CNTLP.COM
*CALL ENGP.COM
*CALL EULCS.COM
*CALL FNMP.COM
*CALL FUNCS.COM
*CALL FUSP.COM
*CALL GPARAM.COM
*CALL RCOEFF.COM
*CALL ROTP.COM
*CALL SHIPP.COM
*CALL TAILP.COM
C**************************************
C
C*   THE FOLLOWING DATA IS FOR COBRA DATA SOURCE IS TECHNICAL REPORT
C*   ECOM-0387-F2A
C
C*   / ADVP   /--AERODYNAMIC PARAMETERS
C
      DATA ADVP1   / 3.28E-4 /
      DATA ADVP2   / 3.0     /
C
C*   / CNTL P  /--CONTROL STICK PARAMETERS
C
      DATA AOSC0  /  8.5  /
      DATA A1SC0  / -10.8 /
      DATA B1SC0  / -13.2 /
      DATA THTRC0 / 23.0  /
      DATA XAOSG  /  0.92 /
      DATA YCSG   /  1.85 /
      DATA XCSG   /  2.81 /
      DATA XTRG   / - 7.00 /
      DATA XAOSR  /  13.3 /
      DATA YCSR   /  9.6  /
      DATA XCSR   /  9.6  /
      DATA XTRR   /  6.0  /
C
C*   / ENG P   /--ENGINE PARAMETERS
C
      DATA IROT    / 2820.0 /
      DATA CKE1    / 10529.0 /
C
C*   / EUL CS  /--EULER COSINE AND SINES
C
      DATA COSFI  / 1.0   /
      DATA SINFI  / 0.0   /
```

```
      DATA COSTH  / 1.0   /
      DATA SINTH  / 0.0   /
      DATA COSSI  / 1.0   /
      DATA SINSI  / 0.0   /
C
C*   / F N M P /--PARAMETERS FOR CALCULATION OF FORCES AND MOMENTS
C
      DATA DX     / -0.5125 /
      DATA DXHS   / 16.55  /
      DATA DXTR   / 26.75  /
      DATA DXVS   / 25.08  /
      DATA DZ     / 7.584  /
      DATA DZTR   / 2.863  /
      DATA DZVS   / 4.40   /
      DATA DXW    / -0.67  /
      DATA DZW    / 7.554  /
      DATA IR     / 0.00   /
C
C*   / FUS P  /--FUSELAGE PARAMETERS
C
      DATA CLF2   / -20.0  /
      DATA CMF2   /  0.0   /
      DATA CNF2   / -30.0  /
      DATA CKRF   /  0.5   /
      DATA CKWIWM /  0.0   /
      DATA IW     /  0.244 /
      DATA VLF    / 100.0  /
      DATA VMF    / 100.0  /
      DATA VNF    / 100.0  /
      DATA SXF1   /  10.0  /
      DATA SXF2   /  10.0  /
      DATA SYF    /  10.0  /
      DATA SZF    /  10.0  /
      DATA SXW    /  27.8  /
      DATA SZW    /  27.8  /
C
C*   / GPARAM  /--GENERAL PARAMETERS
C
      DATA DELT   /  0.03     /
      DATA RHO    /  0.00238  /
      DATA G      / 32.2      /
      DATA PI     /  3.1415926536 /
      DATA PIINV  /  0.31831  / OM
      DATA RAD2DEG / 57.2957795 /
      DATA DEG2RAD /  0.01745329 /
C
C*   / RCOEFF  /--MAIN ROTOR ELEMENTAL LIFT AND DRAG COEFFICIENTS
C       C - IS FOR 3 ELEMENT ROTOR
      DATA CKL    /  5.32,  4.95,  4.35, 17*0. /
      DATA CKD    /  5.32,  4.95,  4.35, 17*0. /
      DATA CKQ    / 52.68, 70.79, 92.57, 17*0. /
      DATA CKDL   /  5.32,  4.95,  4.35, 17*0. /
      DATA CKQL   / 52.68, 70.79, 92.57, 17*0. /
      DATA CKM    / 52.68, 70.79, 92.57, 17*0. /
      DATA Y      /  9.90, 14.30, 18.70, 17*0. /
      DATA WF     /  1.00,  1.00,  1.00, 17*0. /
C
C*   / ROT P  /--MAIN ROTOR PARAMETERS
C
      DATA TWIST  / -0.175 /
      DATA FMRS   /  0.0   /
      DATA CKAO   /  7.04E-4 /
      DATA CKAB1  /  7.04E-4 /
```

```
      DATA CKAB2  / 1.0   /
      DATA CKAB3  / 0.0   /
      DATA NRAD   / 3     /
      DATA NAZ    / 4     /
      DATA NB     / 2     /
      DATA RB     / 22.0  /
      DATA EFH    / 0.0   /
C
C*   / SHIP P /--INERTIA AND MASS PARAMETERS
C
      DATA IXX    / 2530.0 /
      DATA IYY    / 11716.0 /
      DATA IZZ    / 10164.0 /
      DATA MASS   / 274.0 /
C
C*   / TAIL P /--TAIL ROTOR AND STABILIZER PARAMETERS
C
      DATA CKTR1  /  1.0  /
      DATA CKTR2  / 4000.0 /
      DATA IHS    /  0.0  /
      DATA IVS    / - 0.0785 /
      DATA SZHS   /  13.5  /
      DATA SYVS   /  22.5  /
C
C*    INTERNAL DATA FOR FUNCTION GENERATION
C
      DATA (D001(K),K=2,9) / -.26, 3.12, 3.12, .26, 3.12, .26, 0.0, 0.0 /
      DATA (D002(K),K=2,9) / .784, 1., 0., .049, 2.672, .216,  3.59, .27 /
      DATA (D003(K),K=2,9) / -.26, 0., 3.12, .26, 3.12, .26, 0., 0. /
      DATA (D004(K),K=2,9) / -.26, 0., 3.12, .26, 3.12, .26, 0., 0. /
      DATA (D005(K),K=2,9) / -.26, 0., 3.12, .26, 3.12, .26, 0., 0. /
      DATA (D006(K),K=2,9) / .784, 1., 0., .049, 2.672, .216,  3.59, .27 /
      DATA (D007(K),K=2,9) / -.26, 0., 3.12, .26, 3.12, .26, 0., 0. /
      DATA (D008(K),K=2,9) / .784, 1., 0., .049, 2.672, .216,  3.59, .27 /
      DATA (D009(K),K=2,9) / -.26, 0., 3.12, .26, 3.12, .26, 0., 0. /
      DATA (D010(K),K=2,9) / 1.6, 2.22, 0., .1, 8.32, .62, 1.35, .21 /
      DATA (D011(K),K=2,9) / .471, .51, 0., .471, -.432, .039,-.33, .09 /
      DATA (D013(K),K=2,9) / 14., 34., 0., 7., 26., 20., 320., 118. /
      DATA (D014(K),K=2,9) / 33., 100., 0., 33., 34., 67., 100., 100. /
      DATA (D015(K),K=2,9) / 10., 50., 0., 10., 10., 20., 0., 0. /

      DATA (D016(K),K=2,18) / .50605, .50605, .50605, .0349, 0., 0., 0.,
     *                0., 35., 1100., 1100., -110., 110., 0.,  0., 0., 0. /

      DATA (D017(K),K=2,18) / .50605, .50605, .50605, .0349, 0., 0., 0.,
     *                0., 35., 1100., 1100., -110., 110., 0.,  0., 0., 0. /

      DATA (D018(K),K=2,27) / -.092, .084, .353, .261, .18, .088, .96,
     *                .348, 7., 20., 20., 20., 20., 0., 0., 0.,
     *                 0., 3., 360., 360., 0., 120., 0., 0., 0.,  0. /
C
C*    AERO DATA TAPE DATA STATEMENTS (USED TO BE ON TAPE40)
C
      DATA F001 /
     * -.020385 , -.28538  , -.54407  , -.72308  , -.79846  , -.75846  , -.71904  ,
     * -.69404  , -.66849  , -.62618  , -.42655  , -.12774  , .166    , .46371  ,
     *  .73487  , .85767   , .96585   , .9604    , .95042   , .84925   , .74405  ,
     *  .53808  , .34481   , .17769   , .01269   , .000     , .000     , .000    /

      DATA F002 /
     * -1.040   ,-1.054   ,-1.068   ,-1.101   ,-1.1371  , -1.1836  ,-1.2479  ,
     * -1.3121  ,-1.318   ,-1.3158  ,-1.3135  ,-1.3113  , -1.309   ,-1.3068  ,
     * -1.3045  ,-1.3023  ,-1.300   ,-1.0809  , -.8188  , -.42778  , .820    ,
```

```
* 1.090  , 1.320  , 1.320  , 1.2007  , 1.0841  , .000  , .000  /

 DATA F003 /
* .23077 , 3.2308  , 6.000  , 6.000  , 6.000  ,  6.000  , 6.000  ,
* 6.000  , 6.000  , 6.000  , 5.9793  , 4.1679  , .000  , .000  /

 DATA F004 /
* .071154 , .99615 , 1.9352  , 3.0327  , 3.8529  , 3.9279  , 3.9971  ,
* 3.9221  , 3.8179  , 2.9781  , 1.8379  , .79389 , .000  , .000  /

 DATA F005 /
* -.18846 ,-2.6385  ,-5.0296  ,-6.6927  ,-7.7777  , -8.2377  ,-8.665  ,
* -8.275  ,-7.8627  ,-6.6058  ,-4.9015  ,-2.4313  ,.000  , .000  /

 DATA F006 /
*  .000  , -.023333 , -.046667 , -.10439 , -.16785 , -.26436 , -.41779 ,
*  -.57121 , -.590  , -.590  , -.590  , -.590  , -.590  , -.590  ,
*  -.590  , -.590  , -.590  , -.44234 , -.23112 , -.026556 , .17508 ,
*  .37654 , .590  , .590  , .34484 , .09968 , .000  , .000  /

 DATA F007 /
*  .23846 , 3.3385  , 6.2593  , 7.0238  , 7.6349  , 8.0144  , 8.1927  ,
*  7.9973  , 7.6468  , 6.7674  , 5.0648  , 2.8779  , .000  , .000  /

 DATA F008 /
*  -.005  , .0066667, .018333 , .044098 , .072213 , .12246 , .21082 ,
*  .29918 , .34846 , .39229 , .43612 , .47995 , .52378 , .56761 ,
*  .61144 , .65528 , .69911 , .61887 , .3624  , .036667 , -.27692 ,
*  -.58846 , -.700  , -.600  , -.13508 , .32984 , .000  , .000  /

 DATA F009 /
*  .050  , .700  , 1.3185  , 1.5462  , 1.4442  , 1.2962  , 1.2038  ,
*  1.3058  , 1.4536  , 1.5452  , 1.2948  , .84351 , .000  , .000  /

 DATA F010 /
*  .000  , -.24333 , -.48667 , -.730  , -.7925  , -.855  , -.9175  ,
*  -.930  , -.930  , -.92676 , -.89437 , -.86197 , -.82958 , -.79718 ,
*  -.76479 , -.73239 , -.700  , -.930  , -.930  , -.890  , -.5162  ,
*  -.1424  , .000  , .000  , .000  , .000  , .000  , .000  /

 DATA F011 /
*  .000  ,-1.410  ,-1.4064  ,-1.398  ,-1.3896  , -1.3812  ,-1.3728  ,
* -1.354  ,-1.330  ,-1.306  ,-1.282  ,-1.258  , -1.234  ,-1.210  ,
* -1.2291  ,-1.2482  ,-1.2674  ,-1.2865  ,-1.3056  , -1.3247  ,-1.3439  ,
* -1.363  ,-1.3762  ,-1.3422  ,-1.3081  ,-1.2741  , -1.240  ,-1.0026  ,
*  -.76511 , -.52766 , -.29021 , -.052766 , .000  , .000  , .000  /

 DATA F013 /
* 1.000  , .700  , .340  , .160  , .000  , .000  , .000  /


 DATA F014 /
* 1.000  , .400  , .000  , .000  , .000  , .000  , .000  /

 DATA F015 /
* 1.000  , .500  , .125  , .000  , .000  , .000  , .000  /

 DATA (F016(K,1),K=1,35) /
*  .622  , .563  , .508  , .455  , .406  , .361  , .130  ,
*  .062  , .013  , .011  , .010  , .009  , .008  , .008  ,
*  .008  , .008  , .008  , .008  , .009  , .010  , .011  ,
*  .013  , .062  , .130  , .361  , .406  , .455  , .508  ,
*  .563  , .622  , .000  , .000  , .000  , .000  , .000  /
```

```
DATA (F016(K,2),K=1,35) /
* .622  , .563  , .508  , .455  , .406  , .361  , .130  ,
* .062  , .013  , .011  , .010  , .009  , .008  , .008  ,
* .008  , .008  , .008  , .008  , .009  , .010  , .011  ,
* .013  , .062  , .130  , .361  , .406  , .455  , .508  ,
* .563  , .622  , .000  , .000  , .000  , .000  , .000  /
 DATA (F016(K,3),K=1,35) /
* .622  , .564  , .508  , .456  , .406  , .361  , .130  ,
* .062  , .013  , .011  , .010  , .009  , .008  , .008  ,
* .008  , .008  , .008  , .008  , .009  , .010  , .011  ,
* .013  , .062  , .130  , .361  , .406  , .456  , .508  ,
* .564  , .622  , .000  , .000  , .000  , .000  , .000  /
 DATA (F016(K,4),K=1,35) /
* .623  , .565  , .509  , .456  , .407  , .361  , .180  ,
* .112  , .044  , .011  , .010  , .009  , .008  , .008  ,
* .008  , .008  , .008  , .008  , .009  , .010  , .011  ,
* .044  , .112  , .180  , .361  , .407  , .456  , .509  ,
* .565  , .623  , .000  , .000  , .000  , .000  , .000  /
 DATA (F016(K,5),K=1,35) /
* .626  , .567  , .511  , .457  , .407  , .361  , .280  ,
* .212  , .144  , .076  , .010  , .009  , .008  , .008  ,
* .008  , .008  , .008  , .008  , .009  , .010  , .076  ,
* .144  , .212  , .280  , .361  , .407  , .457  , .511  ,
* .567  , .626  , .000  , .000  , .000  , .000  , .000  /
 DATA (F016(K,6),K=1,35) /
* .632  , .571  , .514  , .459  , .409  , .362  , .340  ,
* .312  , .244  , .176  , .108  , .041  , .008  , .008  ,
* .008  , .008  , .008  , .008  , .041  , .108  , .176  ,
* .244  , .312  , .340  , .362  , .409  , .459  , .514  ,
* .571  , .632  , .000  , .000  , .000  , .000  , .000  /
 DATA (F016(K,7),K=1,35) /
* .640  , .578  , .519  , .463  , .411  , .362  , .340  ,
* .340  , .340  , .276  , .208  , .141  ,  .074  , .008  ,
* .008  , .008  , .008  , .074  , .141  , .208  , .276  ,
* .340  , .340  , .340  , .362  , .411  , .463  , .519  ,
* .578  , .640  , .000  , .000  , .000  , .000  , .000  /
 DATA (F016(K,8),K=1,35) /
* .653  , .586  , .527  , .468  , .414  , .363  , .340  ,
* .340  , .340  , .340  , .308  , .241  , .174  , .107  ,
* .041  , .041  , .107  , .174  , .241  , .308  , .340  ,
* .340  , .340  , .340  , .363  , .414  , .468  , .527  ,
* .586  , .653  , .000  , .000  , .000  , .000  , .000  /
 DATA (F016(K,9),K=1,35) /
* .672  , .603  , .538  , .476  , .418  , .365  , .340  ,
* .340  , .340  , .340  , .340  , .340  , .274  , .207  ,
* .141  , .141  , .207  , .274  , .340  ,  .340  , .340  ,
* .340  , .340  , .340  , .365  , .418  , .476  , .538  ,
* .603  , .672  , .000  , .000  , .000  , .000  , .000  /
 DATA (F016(K,10),K=1,35) /
* .699  , .624  , .554  , .487  , .425  , .367  , .340  ,
* .340  , .340  , .340  , .340  , .340  , .340  , .340  ,
* .340  , .340  , .340  , .340  , .340  , .340  , .340  ,
* .340  , .340  , .340  , .367  , .425  , .487  , .554  ,
* .624  , .699  , .000  , .000  , .000  , .000  , .000  /

 DATA (F017(K,1),K=1,35) /
* -.955  , -.962  , -.926  , -.885  , -.841  ,  -.794  , -.840  ,
* -.980  ,-1.120  ,-1.183  , -.967  , -.752  ,  -.537  , -.322  ,
* -.107  , .107  , .322  , .537  , .752  ,  .967  , 1.183  ,
* 1.120  , .980  , .840  , .794  , .841  ,  .885  , .926  ,
* .962  , .995  , .000  , .000  , .000  ,  .000  , .000  /
 DATA (F017(K,2),K=1,35) /
* -.995  , -.963  , -.926  , -.885  , -.841  ,  -.794  , -.822  ,
```

```
* -.928  ,-1.034  ,-1.140  , -.982  , -.764  ,  -.546  , -.327  ,
* -.109  ,  .109  ,  .327  ,  .546  ,  .764  ,   .982  , 1.140  ,
* 1.034  ,  .928  ,  .822  ,  .794  ,  .841  ,   .885  ,  .926  ,
* .963   ,  .995  ,  .000  ,  .000  ,  .000  ,   .000  ,  .000  /
 DATA (F017(K,3),K=1,35) /
* -.996  , -.963  , -.926  , -.885  , -.841  ,  -.794  , -.808  ,
* -.885  , -.963  ,-1.040  ,-1.009  , -.785  ,   -.560  , -.336  ,
* -.112  ,  .112  ,  .336  ,  .560  ,  .785  ,  1.009  , 1.040  ,
* .963   ,  .885  ,  .808  ,  .794  ,  .841  ,   .885  ,  .926  ,
* .963   ,  .996  ,  .000  ,  .000  ,  .000  ,   .000  ,  .000  /
 DATA (F017(K,4),K=1,35) /
* -.997  , -.964  , -.927  , -.886  , -.842  ,   -.794  , -.797  ,
* -.851  , -.904  , -.958  ,-1.012  , -.817  ,  -.583  , -.350  ,
* -.116  ,  .116  ,  .350  ,  .583  ,  .817  ,  1.012  ,  .958  ,
* .904   ,  .851  ,  .797  ,  .794  ,  .842  ,   .886  ,  .927  ,
* .964   ,  .997  ,  .000  ,  .000  ,  .000  ,   .000  ,  .000  /
 DATA (F017(K,5),K=1,35) /
*-1.000  , -.966  , -.929  , -.887  , -.842  ,  -.794  , -.787  ,
* -.822  , -.857  , -.892  , -.927  , -.864  ,  -.617  , -.370  ,
* -.123  ,  .123  ,  .370  ,  .617  ,  .864  ,   .927  ,  .892  ,
* .857   ,  .822  ,  .787  ,  .794  ,  .842  ,   .887  ,  .929  ,
* .966   , 1.000  ,  .000  ,  .000  ,  .000  ,   .000  ,  .000  /
 DATA (F017(K,6),K=1,35) /
*-1.005  , -.970  , -.932  , -.890  , -.844  ,  -.794  , -.779  ,
* -.798  , -.818  , -.837  , -.856  , -.875  ,  -.668  , -.401  ,
* -.133  ,  .133  ,  .401  ,  .668  ,  .875  ,   .856  ,  .837  ,
* .818   ,  .798  ,  .779  ,  .794  ,  .844  ,   .890  ,  .932  ,
* .970   , 1.005  ,  .000  ,  .000  ,  .000  ,   .000  ,  .000  /
 DATA (F017(K,7),K=1,35) /
*-1.013  , -.977  , -.937  , -.893  , -.846  ,  -.795  , -.773  ,
* -.779  , -.785  , -.792  , -.798  , -.804  ,  -.749  , -.449  ,
* -.149  ,  .149  ,  .449  ,  .749  ,  .804  ,   .798  ,  .792  ,
* .785   ,  .779  ,  .773  ,  .795  ,  .846  ,   .893  ,  .937  ,
* .977   , 1.013  ,  .000  ,  .000  ,  .000  ,   .000  ,  .000  /
 DATA (F017(K,8),K=1,35) /
*-1.025  , -.987  , -.945  , -.899  , -.850  ,  -.796  , -.768  ,
* -.764  , -.760  , -.756  , -.752  , -.748  ,  -.744  , -.740  ,
* -.283  ,  .283  ,  .740  ,  .744  ,  .748  ,   .752  ,  .756  ,
* .760   ,  .764  ,  .768  ,  .796  ,  .850  ,   .899  ,  .945  ,
* .987   , 1.025  ,  .000  ,  .000  ,  .000  ,   .000  ,  .000  /
 DATA (F017(K,9),K=1,35) /
*-1.042  ,-1.002  , -.957  , -.908  , -.855  ,  -.798  , -.762  ,
* -.748  , -.733  , -.718  , -.704  , -.689  ,  -.675  , -.466  ,
* -.155  ,  .155  ,  .466  ,  .675  ,  .689  ,   .704  ,  .718  ,
* .733   ,  .748  ,  .762  ,  .798  ,  .855  ,   .908  ,  .957  ,
* 1.002  , 1.042  ,  .000  ,  .000  ,  .000  ,   .000  ,  .000  /
 DATA (F017(K,10),K=1,35) /
*-1.067  ,-1.023  , -.974  , -.920  , -.862  ,  -.801  , -.755  ,
* -.727  , -.699  , -.670  , -.642  , -.614  ,  -.500  , -.300  ,
* -.100  ,  .100  ,  .300  ,  .500  ,  .614  ,   .642  ,  .670  ,
* .699   ,  .727  ,  .755  ,  .801  ,  .862  ,   .920  ,  .974  ,
* 1.023  , 1.067  ,  .000  ,  .000  ,  .000  ,   .000  ,  .000  /

 DATA (F018(K,1,1),K=1,7) /
* -.3253  , -.0337  , .0039  , .0635  , .4818  , .000  , .000  /
 DATA (F018(K,2,1),K=1,7) /
* -.3536  , -.0555  , -.0026  , .0477  , .4556  , .000  , .000  /
 DATA (F018(K,3,1),K=1,7) /
* -.3789  , -.072  , -.0094  , .0267  , .4264  , .000  , .000  /
 DATA (F018(K,1,2),K=1,7) /
* -.4091  , -.0651  , .0452  , .1548  , .6081  , .000  , .000  /
 DATA (F018(K,2,2),K=1,7) /
* -.4561  , -.1155  , -.0056  , .1045  , .5627  , .000  , .000  /
```

```
      DATA (F018(K,3,2),K=1,7) /
     * -.5027  , -.1656  , -.0562  , .0541  , .5168  , .000  , .000  /
      DATA (F018(K,1,3),K=1,7) /
     * -.5328  , -.0933  , .0527  , .1985  , .7829  , .000  , .000  /
      DATA (F018(K,2,3),K=1,7) /
     * -.5919  , -.1532  , -.0074  , .1386  , .7241  , .000  , .000  /
      DATA (F018(K,3,3),K=1,7) /
     * -.651  , -.2131  , -.0672  , .0787  , .6657  , .000  , .000  /
      DATA (F018(K,1,4),K=1,7) /
     * -.6602  , -.1289  , .0547  , .2362  , .9725  , .000  , .000  /
      DATA (F018(K,2,4),K=1,7) /
     * -.7436  , -.1925  , -.0092  , .1744  , .9091  , .000  , .000  /
      DATA (F018(K,3,4),K=1,7) /
     * -.8072  , -.2566  , -.073  , .1105  , .8458  ,  .000  , .000  /
C
      END
```

REFERENCES

1. Houck, Jacob A., Gibson; Lucille H.; and Steinmetz, George G.:
   A Real-Time Digital Computer Program for the Simulation of a Single-rotor Helicopter. NASA TM X-2872, 1974.

2. Callan, William M.; Houck, Jacob A.; and DiCarlo, Daniel J.:
   Simulation Study of Intracity Helicopter Operations Under Instrument Conditions to Category I Minimums. NASA TN D-7786, 1974.

3. Parrish, Russell V.; Houck, Jacob A.; and Martin, Dennis J., Jr.:
   Empirical Comparison of a Fixed-Base and a Moving-Base Simulation of a Helicopter Engaged in Visually Conducted Slalom Runs. NASA TN D-8424, 1977

4. Leslie, Richard A.:
   Reference Manual for sifplot - A Command File Driven Plotting System, (Contract NASI-19119) April, 1993.

5. Grove, Randall D.:
   Real-Time Simulation User's Guide. Central Scientific Computing Complex Document R-1c, January, 1993.

TABLE I.A. - S-61 PROGRAM DAILY STATIC CHECK

STATIC CHECK CASE DATE: 16-Jul-93 TIME: 10:09:09 S61-COBRA

SAS IS OFF

TIME	=	0.00000000E+00	GRWT	=	0.88228000E+04	CG	= 0.19385000E+03
ASPD	=	0.80308977E+02	H	=	0.50000000E+04	HDOT	= -0.45326371E-06
OMEG	=	0.33900000E+02	OMEGDT	=	0.48377378E-14	DELE	= 0.11253091E+00
LMR	=	0.90294960E+04	FXR	=	-0.57512279E+03	QE	= 0.64038741E+04
QMR	=	0.64038741E+04	FYR	=	-0.93507970E+02	WM	= 0.90948165E+01
THTR0	=	0.29313655E-01	YVS	=	0.93209962E+02	VTR	= 0.00000000E+00
YTR	=	0.15021493E+03	ZHS	=	-0.12308727E+02	WHS	= -0.13389733E+02
P	=	0.00000000E+00	Q	=	0.00000000E+00	R	= 0.00000000E+00
PDOT	=	0.11730464E+00	QDOT	=	-0.33939905E-01	RDOT	= -0.28446974E-02
LA	=	0.29678075E+03	MA	=	-0.39763993E+03	NA	= -0.28913505E+02
LF	=	-0.66221811E-06	MF	=	0.41189236E+02	NF	= 0.20280529E-05
U	=	0.13500000E+03	V	=	0.00000000E+00	W	= -0.13180090E+02
UDOT	=	0.16087306E-05	VDOT	=	0.95691213E-07	WDOT	= -0.86299928E-01
XA	=	-0.85717235E+03	YA	=	0.14991692E+03	ZA	= -0.88034286E+04
XF	=	-0.24281747E+03	YF	=	-0.24005813E-06	ZF	= 0.46629397E+03
XW	=	-0.39232093E+02	WWM	=	0.00000000E+00	ZW	= -0.22791781E+03
ALFF	=	-0.13056774E+00	BETF	=	0.00000000E+00	ALFW	= 0.11343226E+00
PHI	=	-0.17073581E-01	THETA	=	-0.97307773E-01	PSI	= 0.00000000E+00
PHIDOT	=	0.00000000E+00	THEDOT	=	0.00000000E+00	PSIDOT	= 0.00000000E+00
AOSS	=	0.44583418E-01	A1SS	=	0.61468875E-01	B1SS	= 0.25262053E-02
AOS0	=	0.23968540E+00	A1S0	=	-0.16621453E-01	B1S0	= -0.11830726E-01

TABLE I.B. - S-61 PROGRAM FULL VALIDATION STATIC CHECK

STATIC CHECK CASE DATE: 16-Jul-93 TIME: 10:09:09 S61-COBRA

SAS IS OFF

TIME	=	0.00000000E+00	GRWT	=	0.88228000E+04	CG	=	0.19385000E+03
ASPD	=	0.80308977E+02	H	=	0.50000000E+04	HDOT	=	-0.45326371E-06
OMEG	=	0.33900000E+02	OMEGDT	=	0.48377378E-14	DELE	=	0.11253091E+00
LMR	=	0.90294960E+04	FXR	=	-0.57512279E+03	QE	=	0.64038741E+04
QMR	=	0.64038741E+04	FYR	=	-0.93507970E+02	WM	=	0.90948165E+01
THTR0	=	0.29313655E-01	YVS	=	0.93209962E+02	VTR	=	0.00000000E+00
YTR	=	0.15021493E+03	ZHS	=	-0.12308727E+02	WHS	=	-0.13389733E+02
P	=	0.00000000E+00	Q	=	0.00000000E+00	R	=	0.00000000E+00
PDOT	=	0.11730464E+00	QDOT	=	-0.33939905E-01	RDOT	=	-0.28446974E-02
LA	=	0.29678075E+03	MA	=	-0.39763993E+03	NA	=	-0.28913505E+02
LF	=	-0.66221811E-06	MF	=	0.41189236E+02	NF	=	0.20280529E-05
U	=	0.13500000E+03	V	=	0.00000000E+00	W	=	-0.13180090E+02
UDOT	=	0.16087306E-05	VDOT	=	0.95691213E-07	WDOT	=	-0.86299928E-01
XA	=	-0.85717235E+03	YA	=	0.14991692E+03	ZA	=	-0.88034286E+04
XF	=	-0.24281747E+03	YF	=	-0.24005813E-06	ZF	=	0.46629397E+03
XW	=	-0.39232093E+02	WWM	=	0.00000000E+00	ZW	=	-0.22791781E+03
ALFT	=	-0.13056774E+00	BETF	=	0.00000000E+00	ALFW	=	0.11343226E+00
PHI	=	-0.17073581E-01	THETA	=	-0.97307773E-01	PSI	=	0.00000000E+00
PHIDOT	=	0.00000000E+00	THEDOT	=	0.00000000E+00	PSIDOT	=	0.00000000E+00
AOSS	=	0.44583418E-01	A1SS	=	0.61468875E-01	B1SS	=	0.25262053E-02
AOS0	=	0.23968540E+00	A1S0	=	-0.16621453E-01	B1S0	=	-0.11830726E-01
XAOS	=	0.56880020E+01	AOSC0	=	0.85000000E+01	XAOSG	=	0.92000000E+00
XCS	=	0.44562809E+01	YCS	=	0.53230599E+01	XTR	=	0.30457788E+01
B1SC0	=	-0.13200000E+02	A1SC0	=	-0.10800000E+02	THTRC0	=	0.23000000E+02
XCSG	=	0.28100000E+01	YCSG	=	0.18500000E+01	XTRG	=	-0.70000000E+01
UTSIV	=	0.49893000E+03	BETSI	=	0.47109654E-01	BDTSI	=	-0.20837960E+01
DPDR	=	-0.20372351E+05	MPDR	=	0.26964463E+08	SLDR	=	0.75878118E+07
LPDR	=	0.15384110E+07	QPDR	=	-0.43483423E+06	SQDR	=	0.53814068E+07
SXDR	=	-0.48329646E+06	LRH	=	0.00000000E+00	GEH	=	0.62500000E-05
SYDR	=	-0.78578126E+05	MRH	=	0.00000000E+00	GEV	=	0.00000000E+00
QV	=	0.22278194E+02	CX1	=	-0.47618367E-10	CYF	=	-0.11068835E-08
QL	=	0.21687750E+02	CX2	=	-0.10899334E+01	CZF	=	0.20930510E+01
CLF1	=	-0.30534201E-09	CYVS	=	0.19101405E+00	CXW	=	0.26447536E-01
CNF1	=	0.93511447E-09	CZHS	=	-0.40926010E-01	CZW	=	-0.36800480E+00
CYTR	=	0.37553732E-01	THTRP	=	0.29313655E-01	VTRP	=	0.00000000E+00

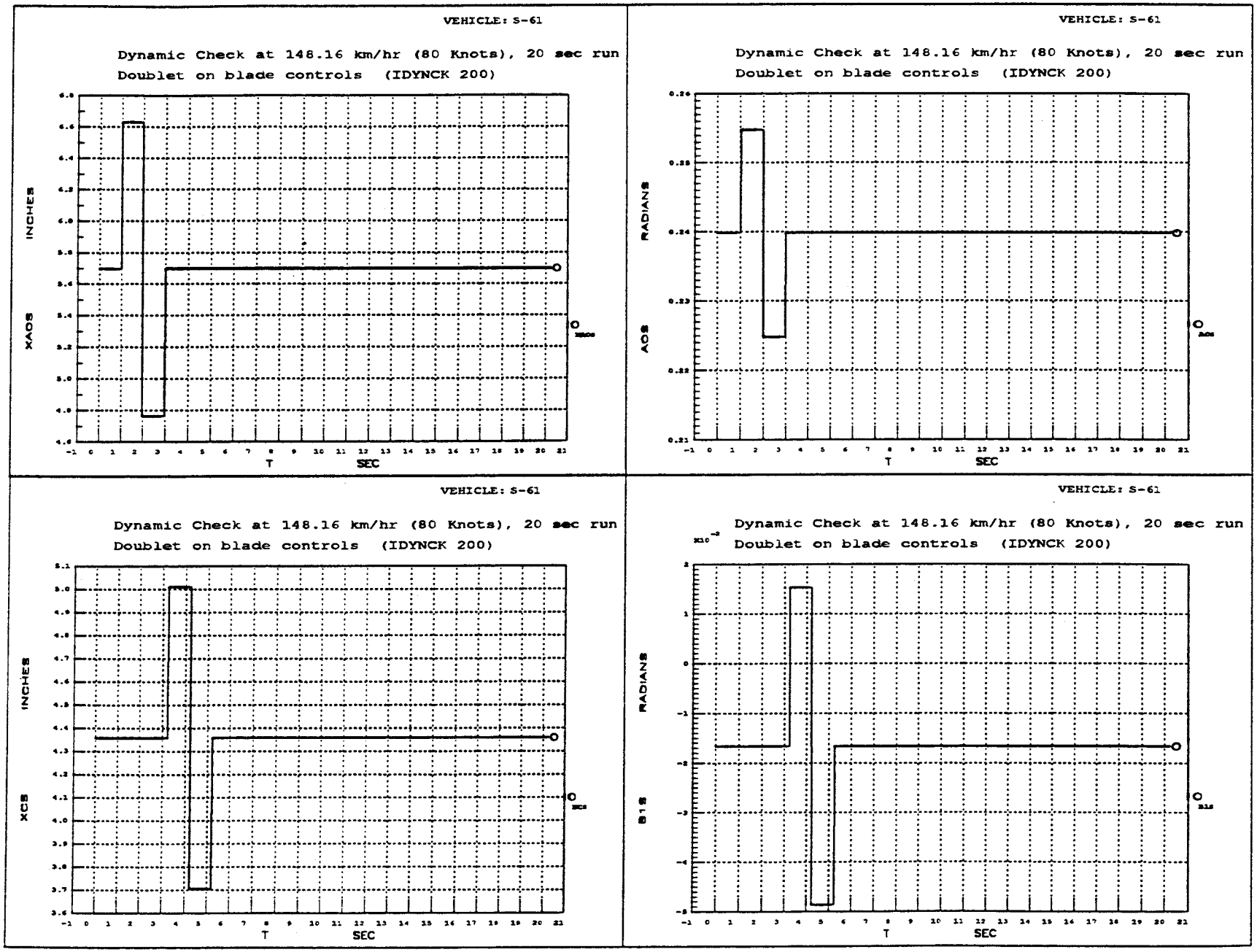


Table II. A. - Dynamic Check Case (Idynck 200)

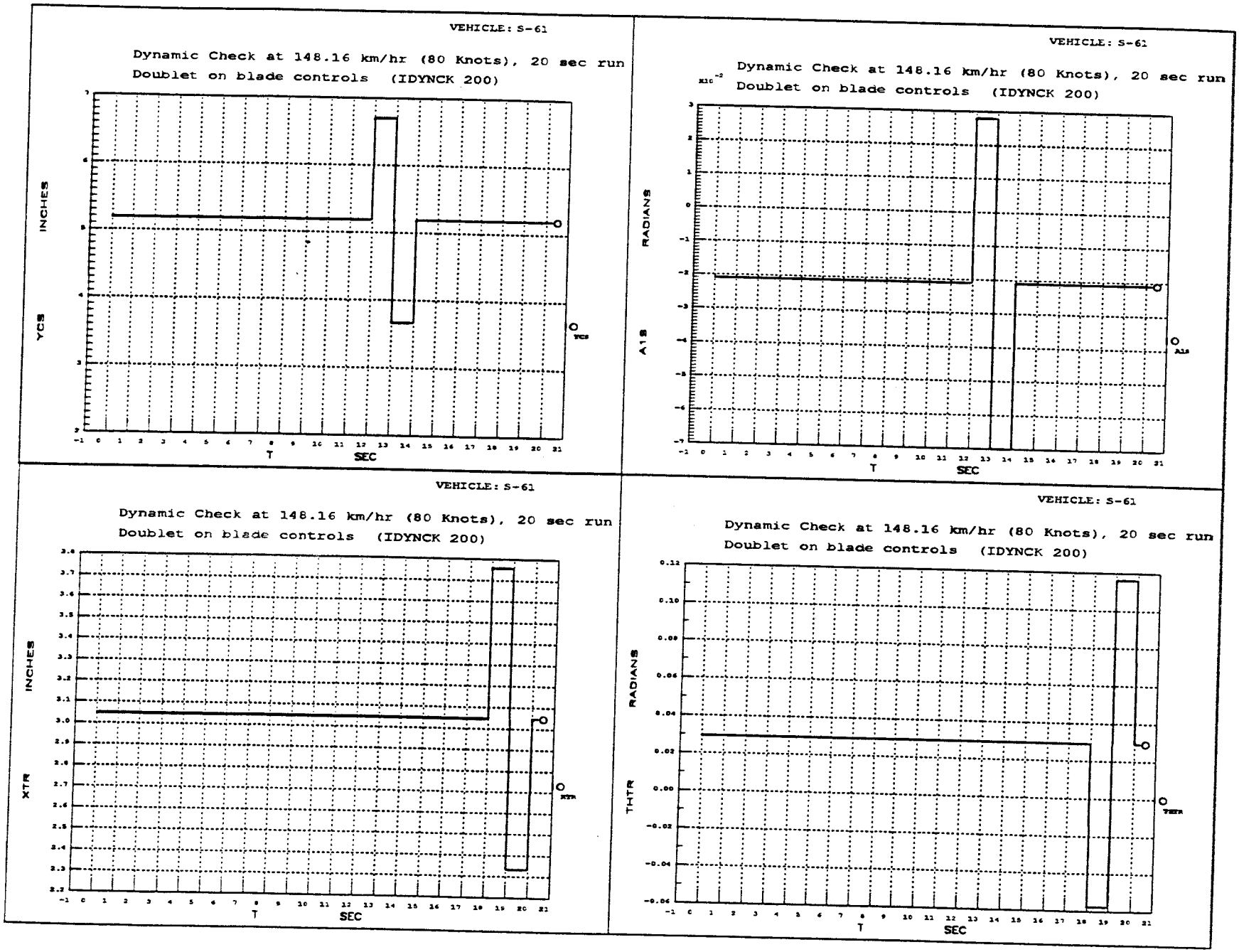


Table II. A. - Continued

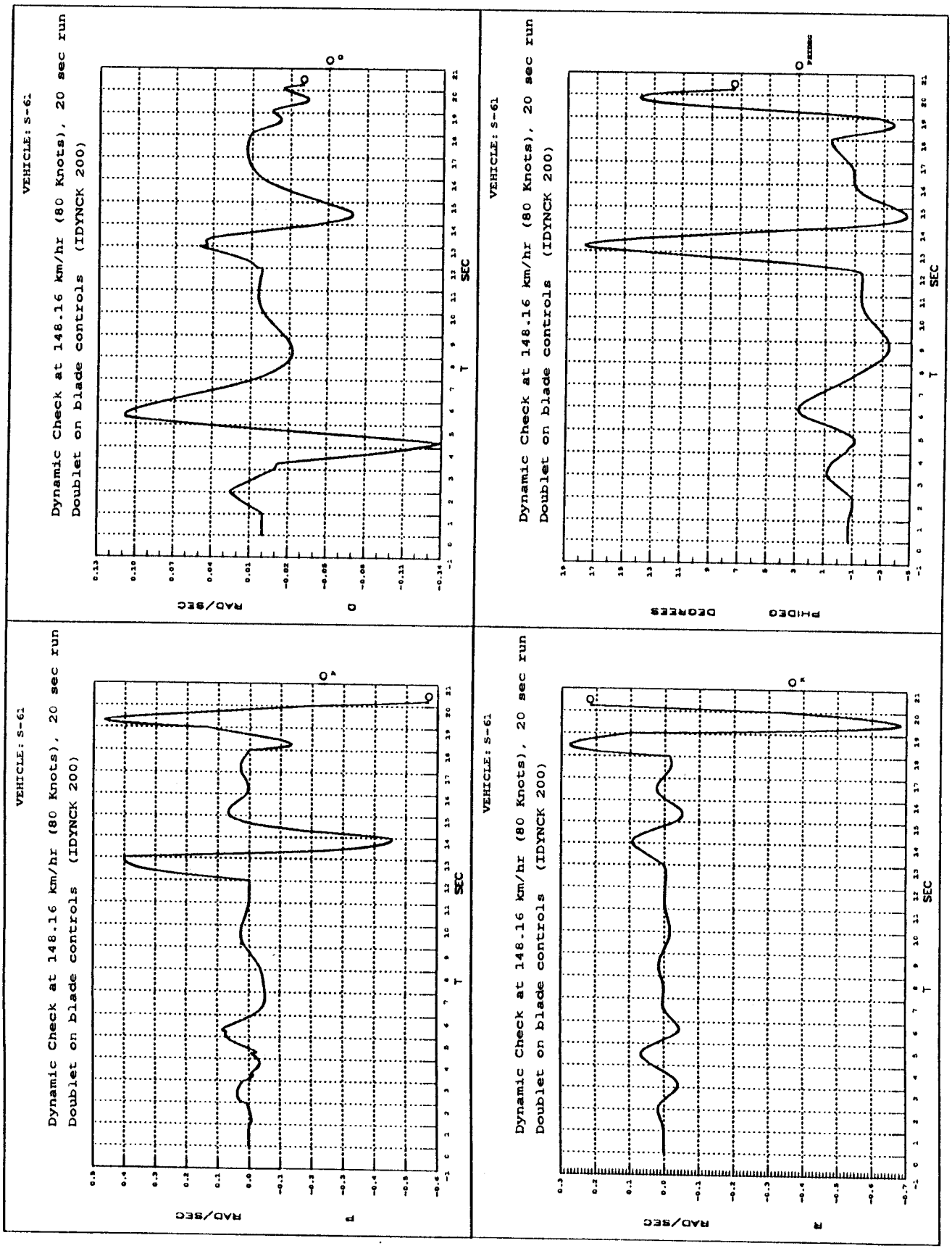


Table II. A. - Continued

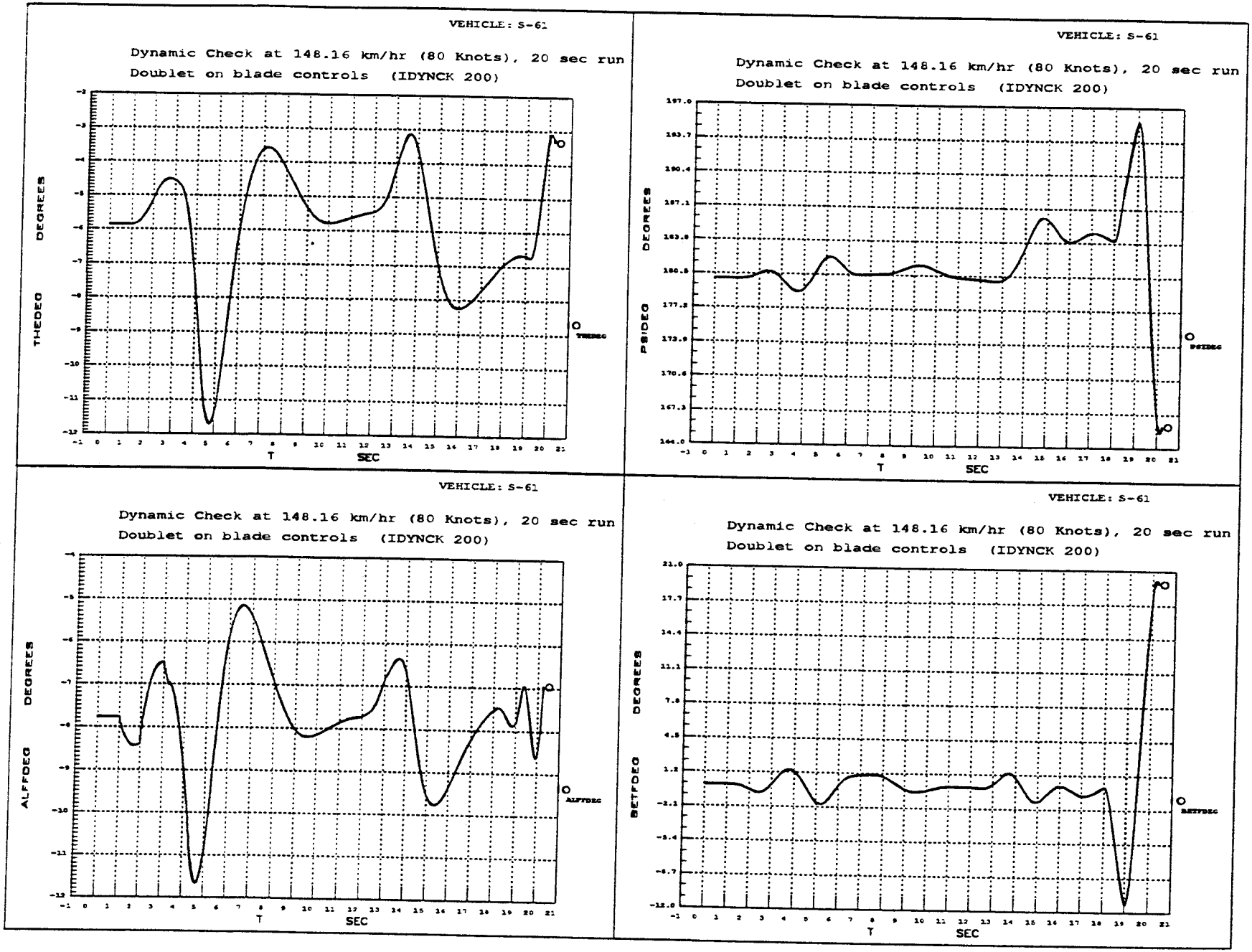


Table II. A. - Continued

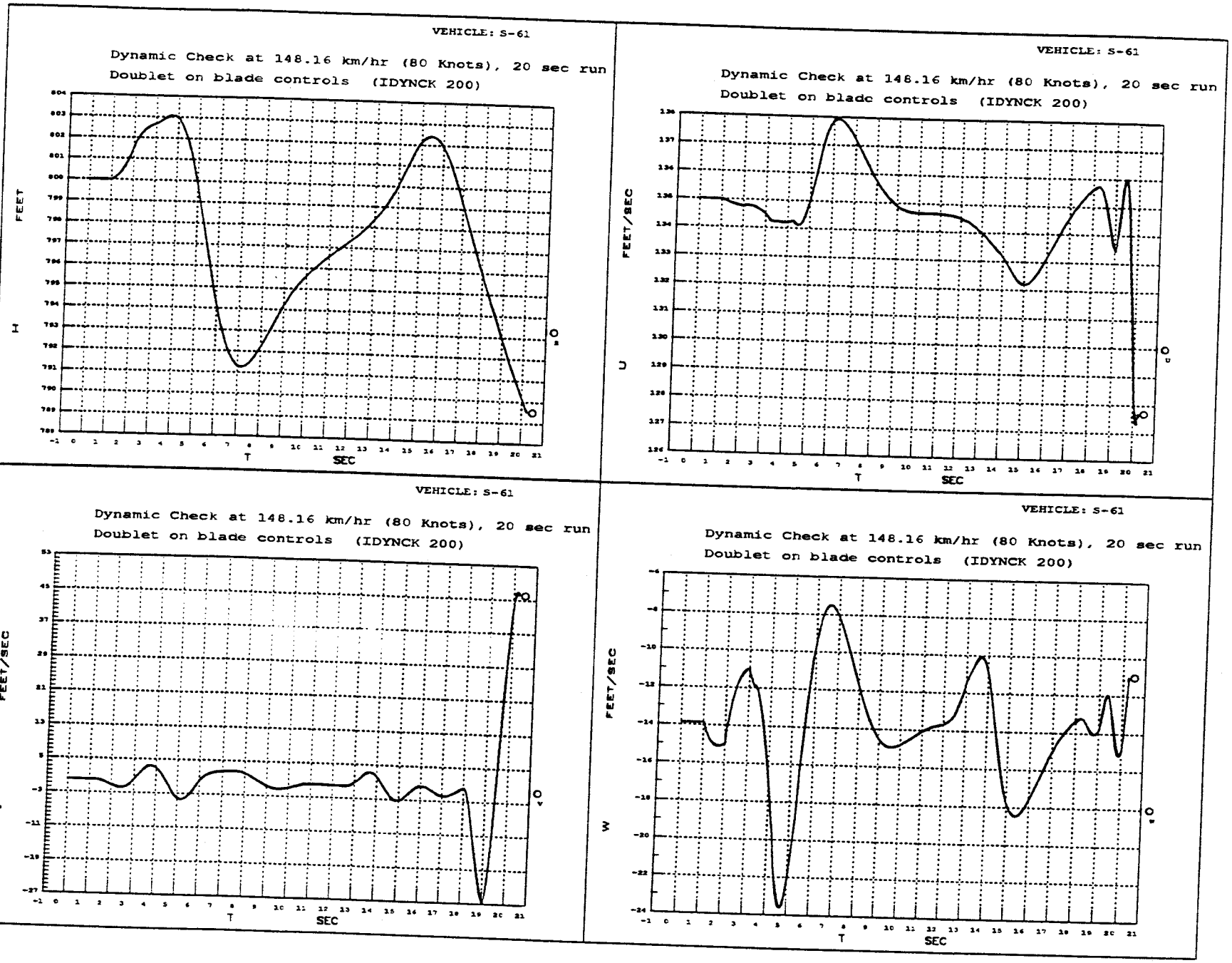


Table II. A. - Continued

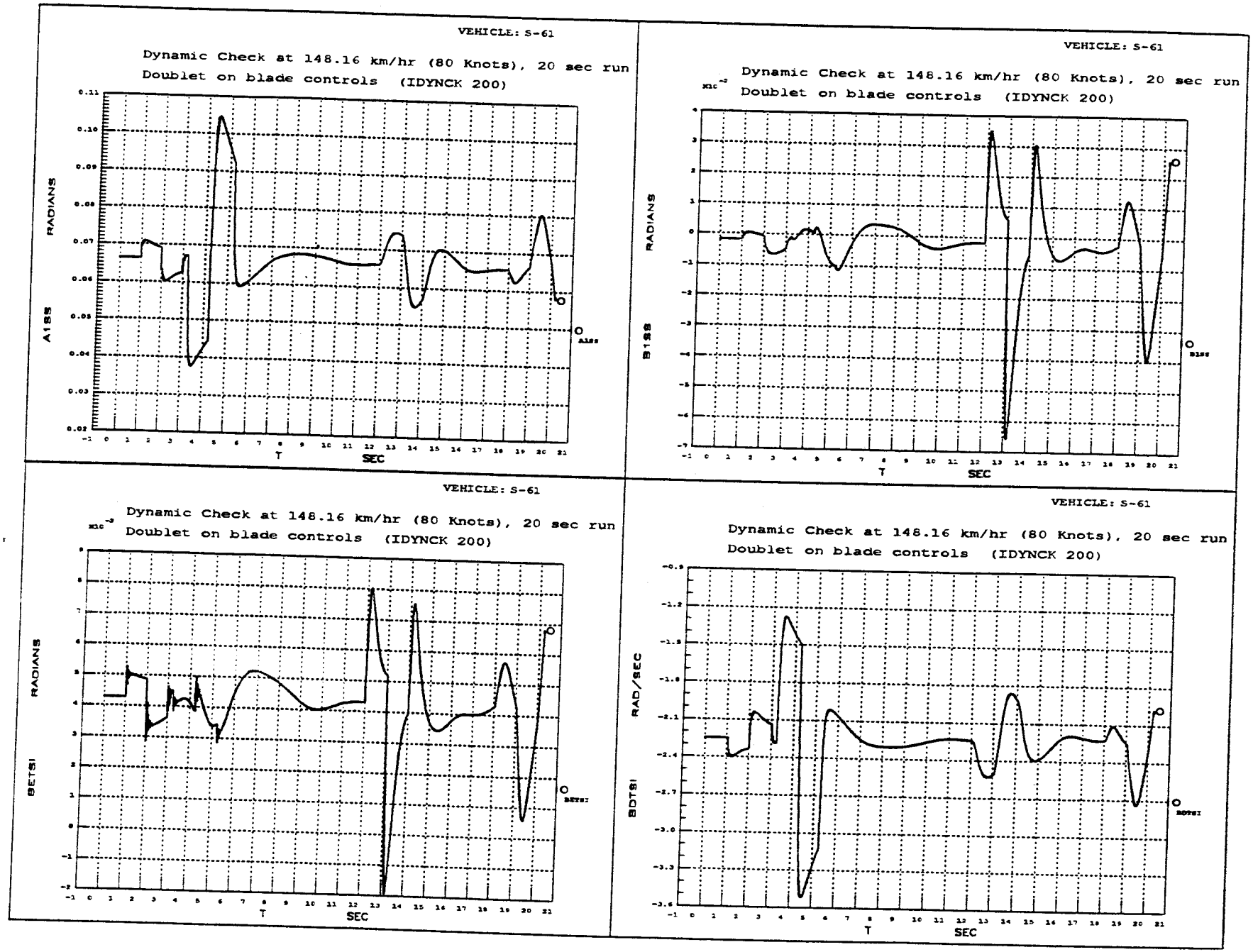


Table II. A. - Concluded

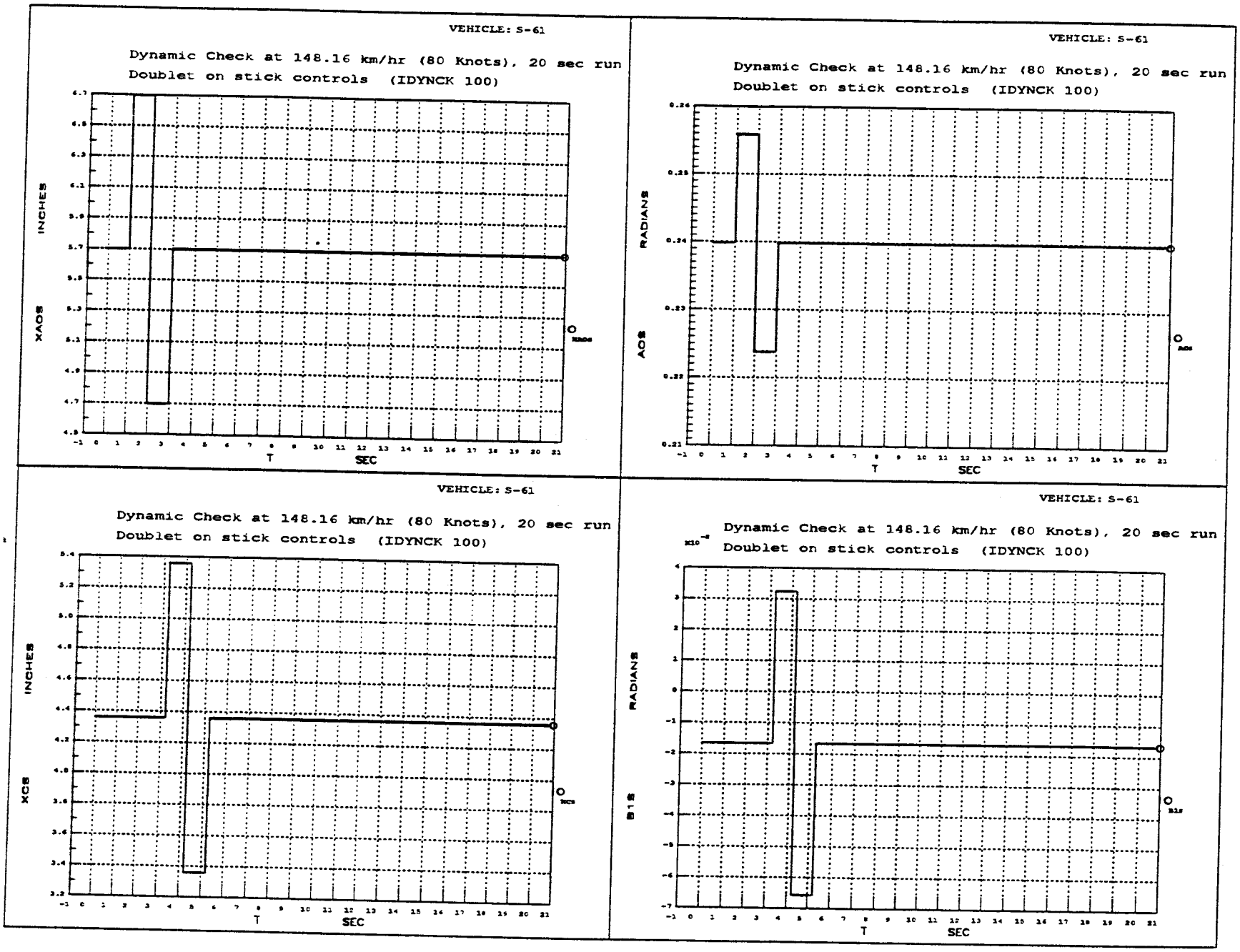


Table II. B. - Dynamic Check Case (Idynck 100)

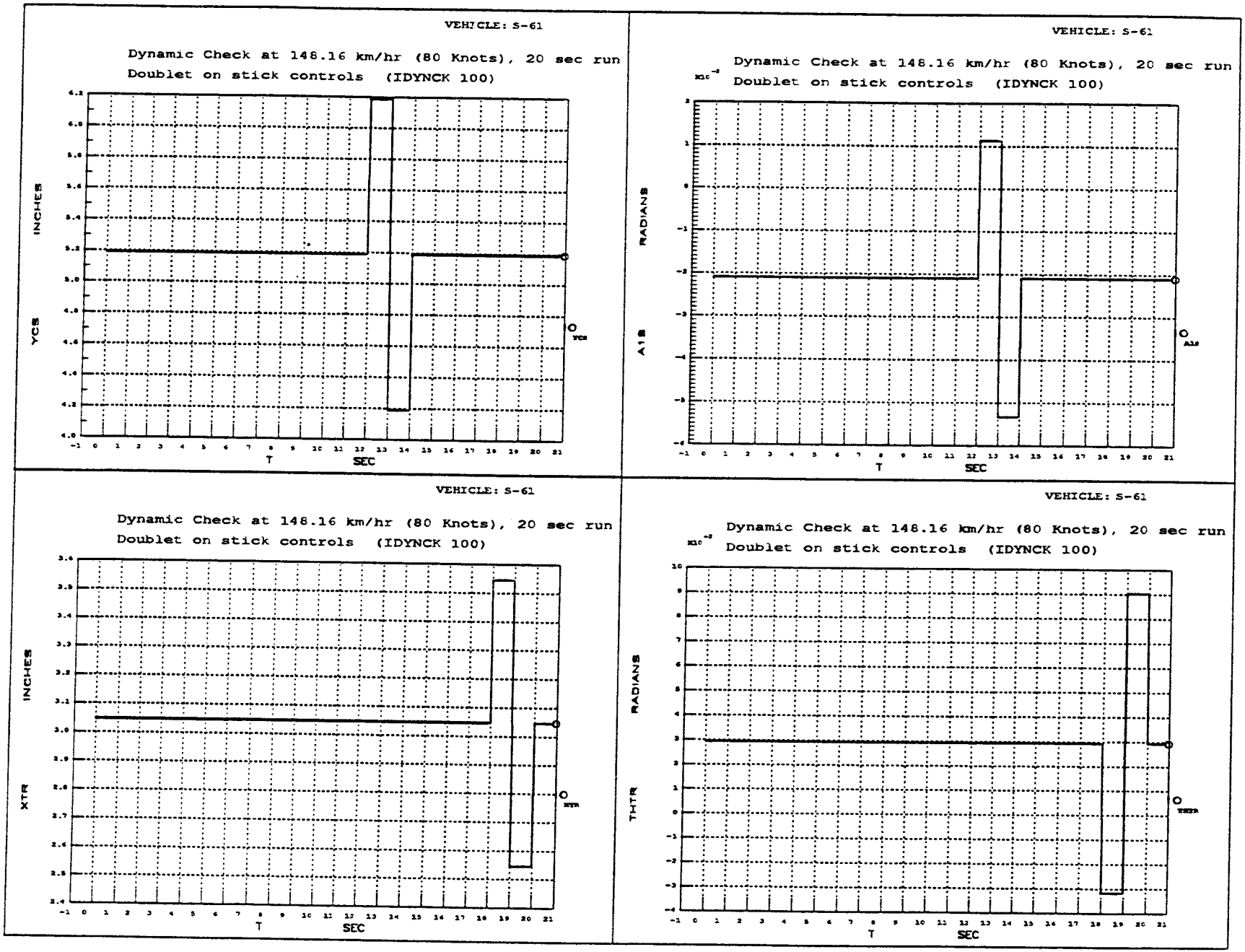


Table II. B. - Continued

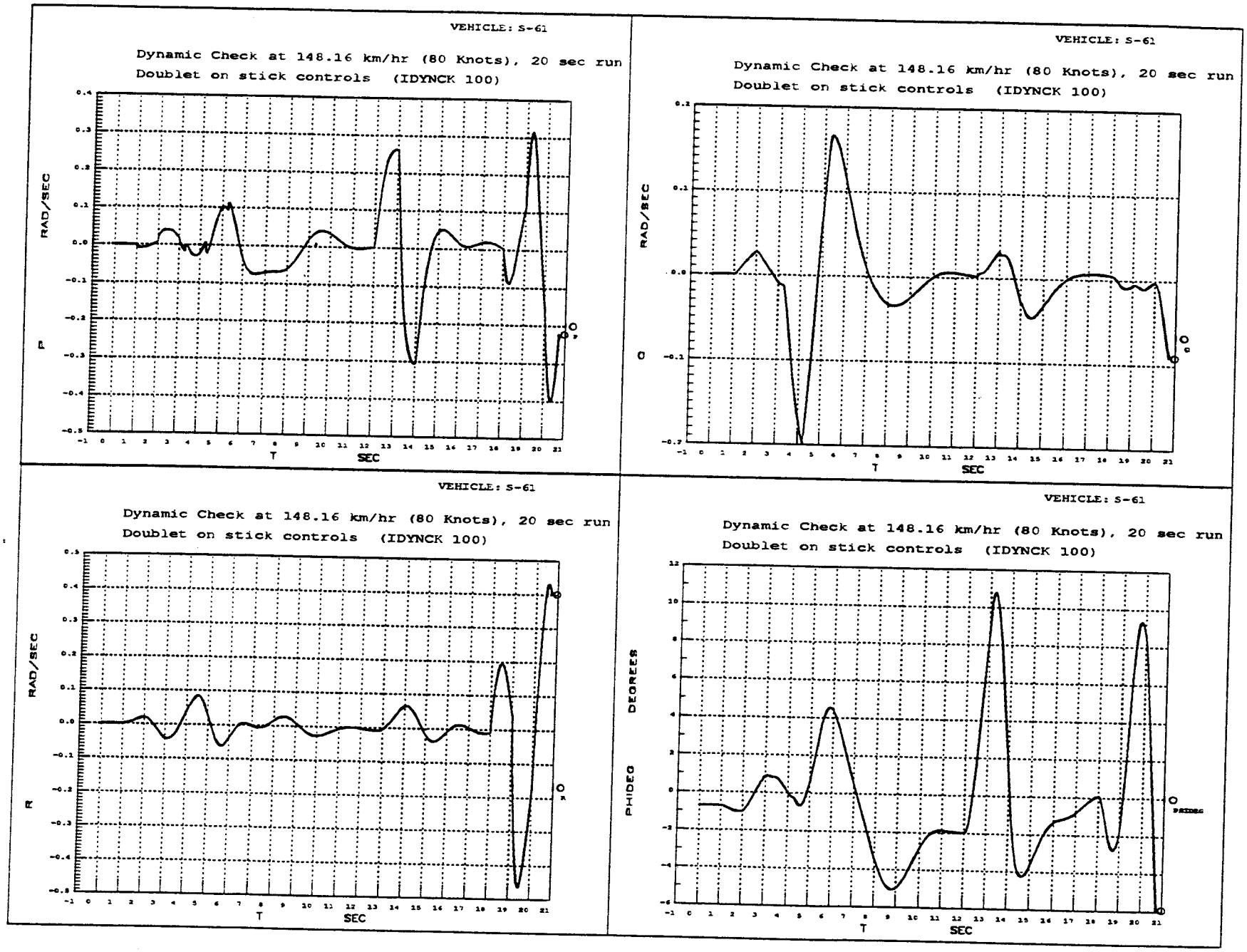


Table II. B. - Continued

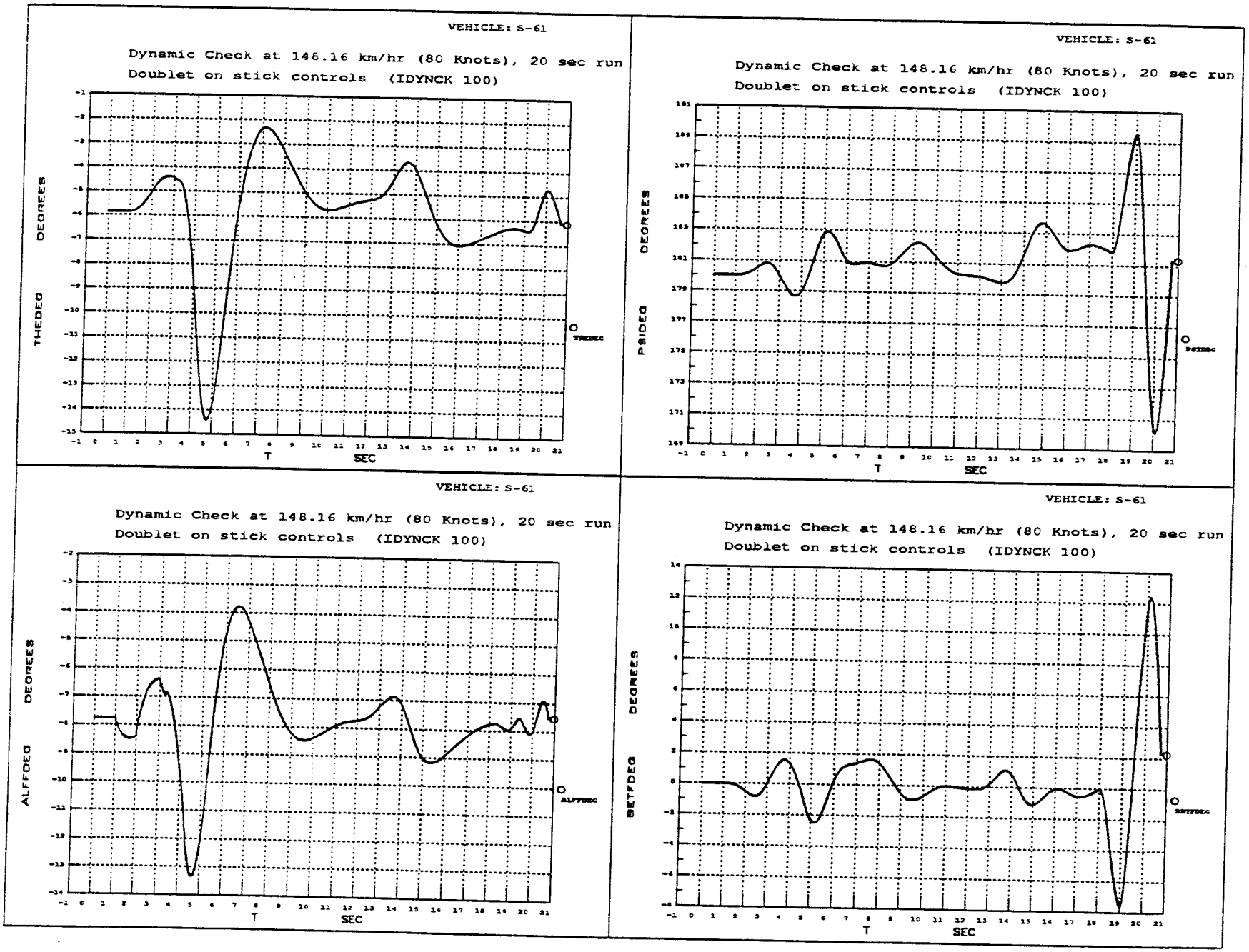


Table II. B. - Continued

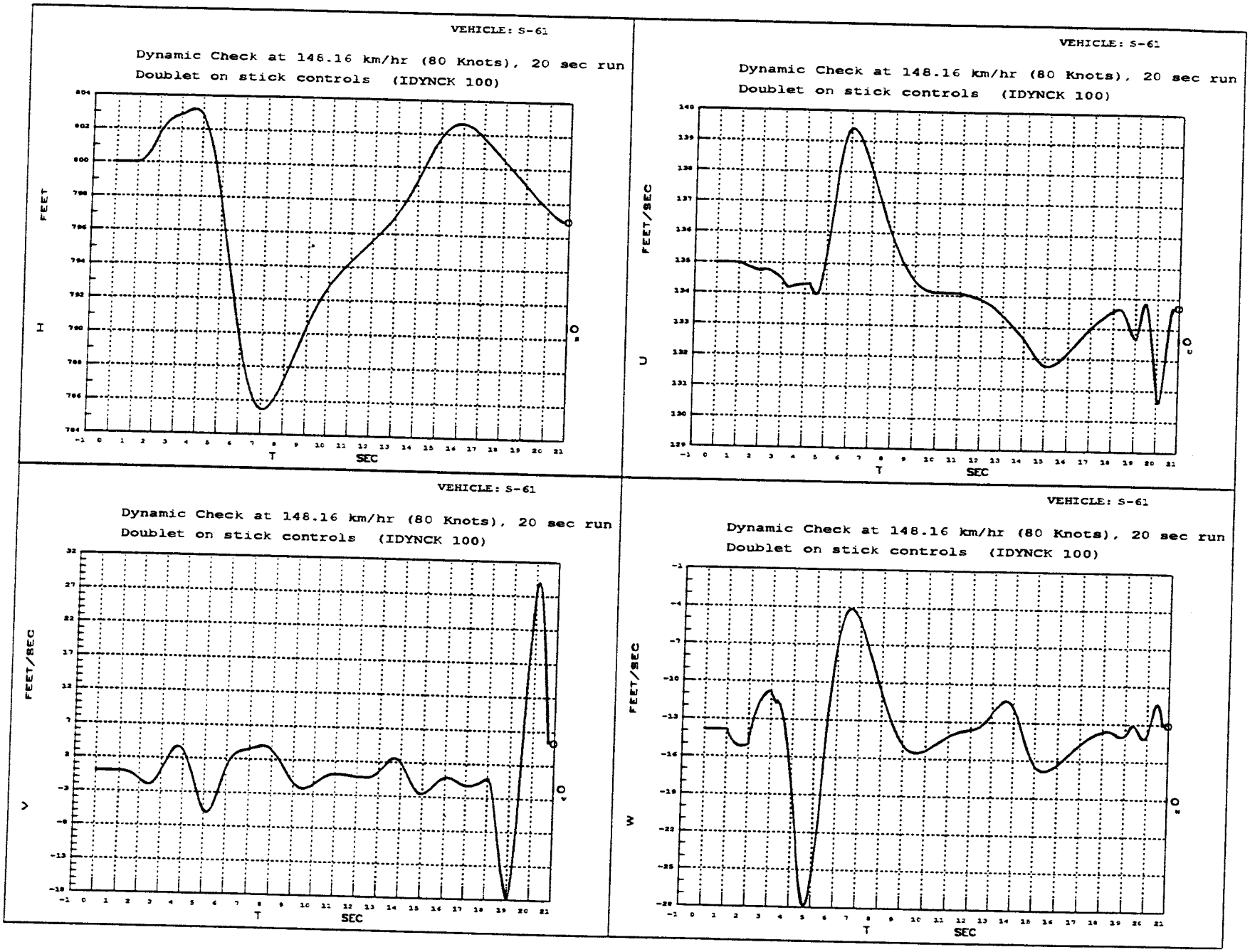


Table II. B. - Continued

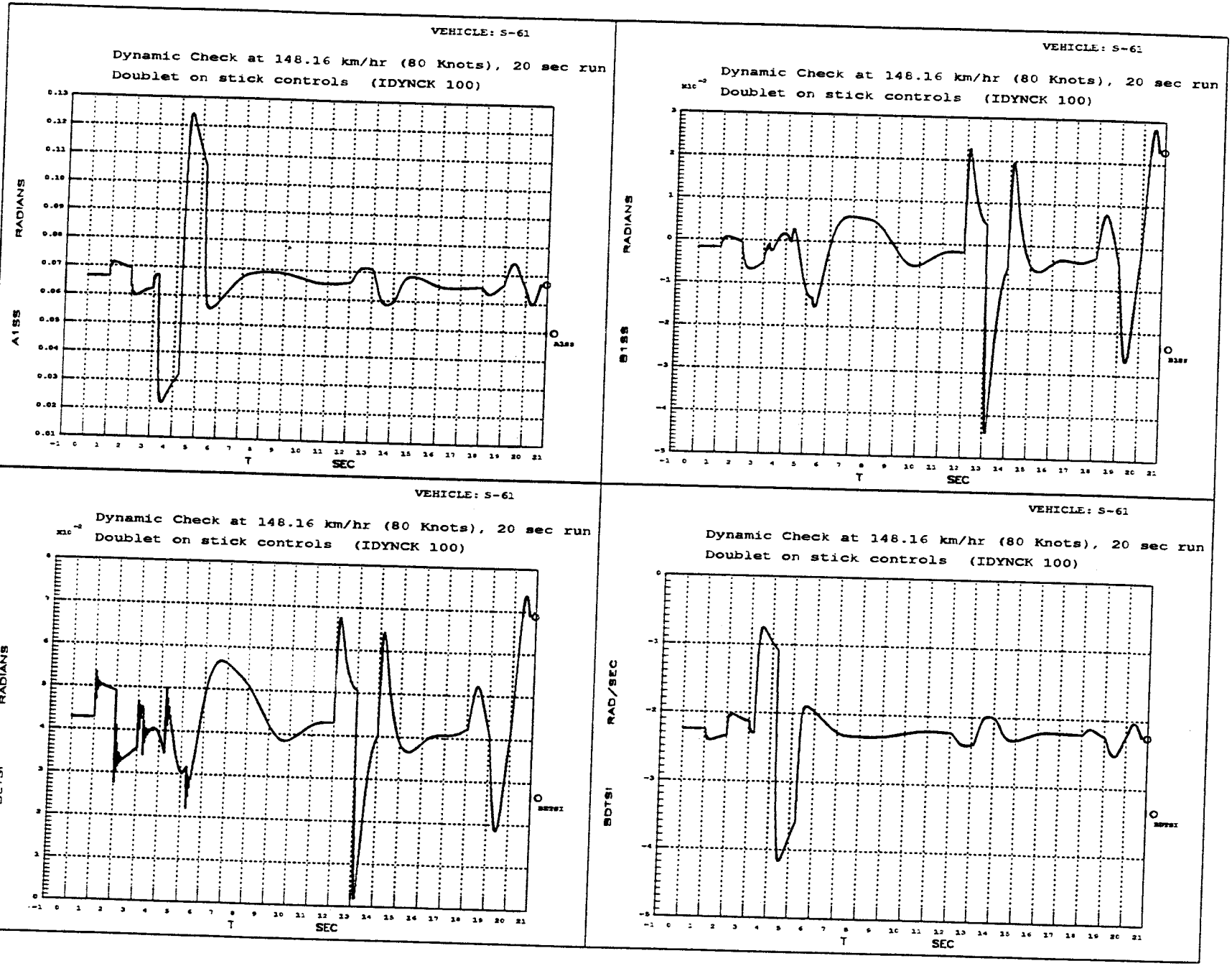


Table II. B. - Concluded

TABLE III.A. - STATIC COCKPIT INSTRUMENT CHECK

DAC			Instrument	
Element	**Variable**	**Voltage**	**Function**	**Reading**
4	BETF	6.99	Slide Slip	4° Right
14 19	PSIDOT XLAMDA	6.00 -1.69	Rate of Turn Indicator Bank Indicator (Right Hand Side of Cockpit)	1 1/2 Needle Width Left 1/2 Ball Width Left
18 19	PSIDOT XLAMDA	-2.99 -1.69	Rate of Turn Indicator Bank Indicator (Left Hand Side of Cockpit)	1 1/2 Needle Width Left 1/2 Ball Width Left
20	ASPD	-3.50	Airspeed Indicator	130 Knots
25 26 27	Alt-Course Sin(h) Cos(h)	-9.74 3.09 9.51	Altimeter	50 feet
28	HDOT	3.73	Vertical Speed	1500 ft/min
31	OMEG	5.41	Main rotor RPM Indicator	65% RPM
32	XAOSL	3.5	Engine Power	50% Power
36	SOUNDSF	5.0	Audio	Rotor blade sound
48	PEDPOS	-0.39	Rudder Trim In	Centered position

TABLE III.B. - CONTROL DEFLECTION CHECK

FORTRAN Variable	Description	Position	Value	Position	Value
XAOS	Collective	Down	0.0	Up	13.3
XCS	Pitch (Longitudinal Stick)	Aft	0.0	Foward	9.6
YCS	Roll (Lateral Stick)	Left	0.0	Right	9.6
XTR	Rudder Pedal	Left	0.0	Right	6.0

TABLE III.C. - CONTROL DEFLECTION VOLTAGE SCALE

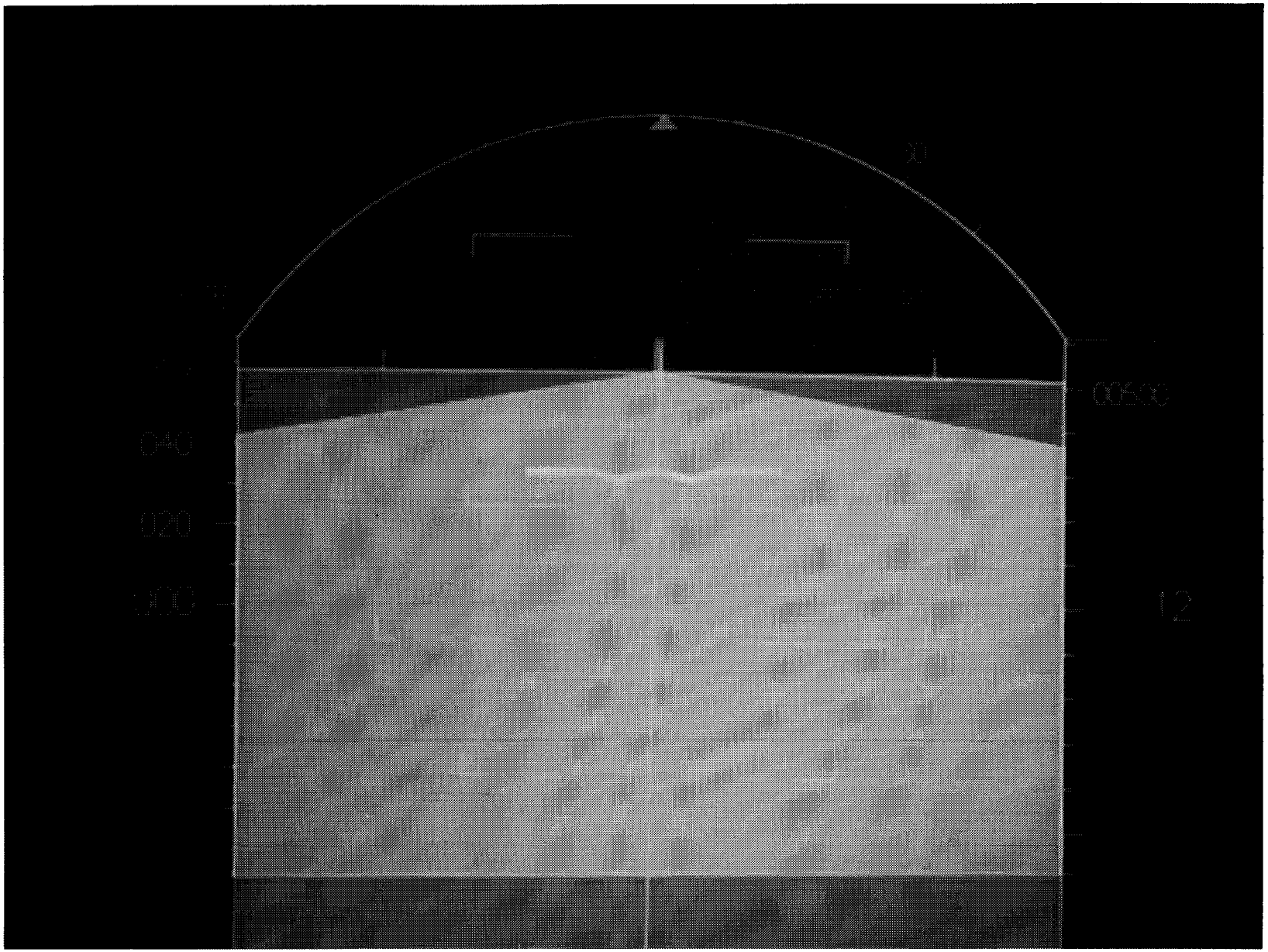
ADC			Cockpit Control	
Element	**Variable**	**Voltage**	**Function**	**Range**
1	X_{CS}	-3.25 to 3.25	Longitudinal cyclic stick	Full forward to full back
2	Y_{CS}	-3.36 to 3.36	Lateral cyclic Stick	Fll left to full right
3	X_{TR}	-7.2 to 7.2	Tail rotor pedals	Full left to full right
13	XTRCOL	-9.37 to 9.37	Tail rotor pedal trim dial	Full left to full right
14	PEDFORC	-1 to +1	Rudder force	Full left to full right
15	X_{AOS}	1.06 to -9.87	Collective Stick	Full down to full up

L-90-13717

Figure 1. Visual Motion Simulator.

L-93-13769

Figure 2. Denver Stapleton Airport data base.



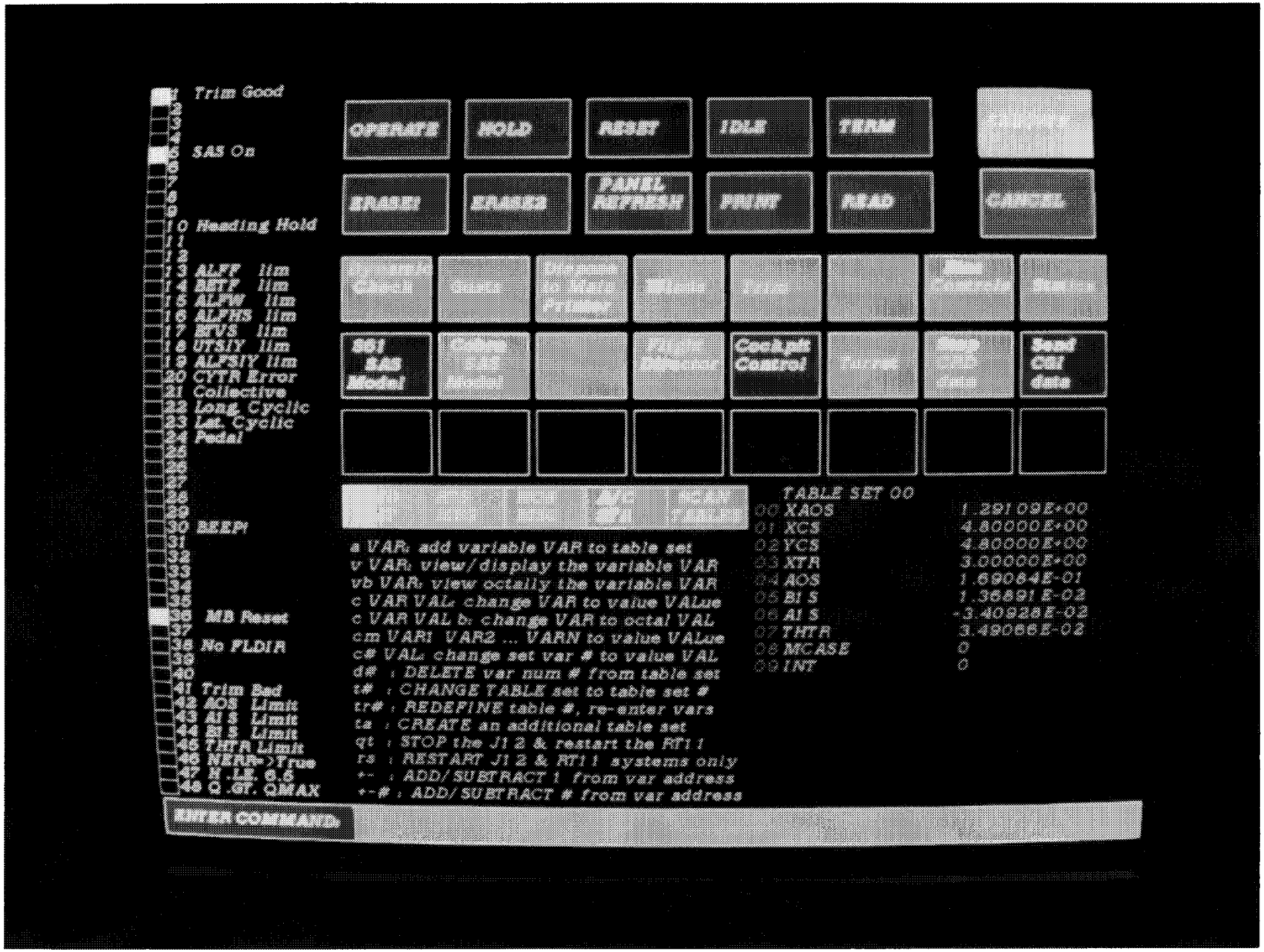
L-93-13770

Figure 3. S-61 electronic attitude directional indicator.

L-94-02093

Figure 4. Real-time control console.

Figure 5. Real-time system console PC.



L-93-13768

Figure 6. S-61 touch screen console format.